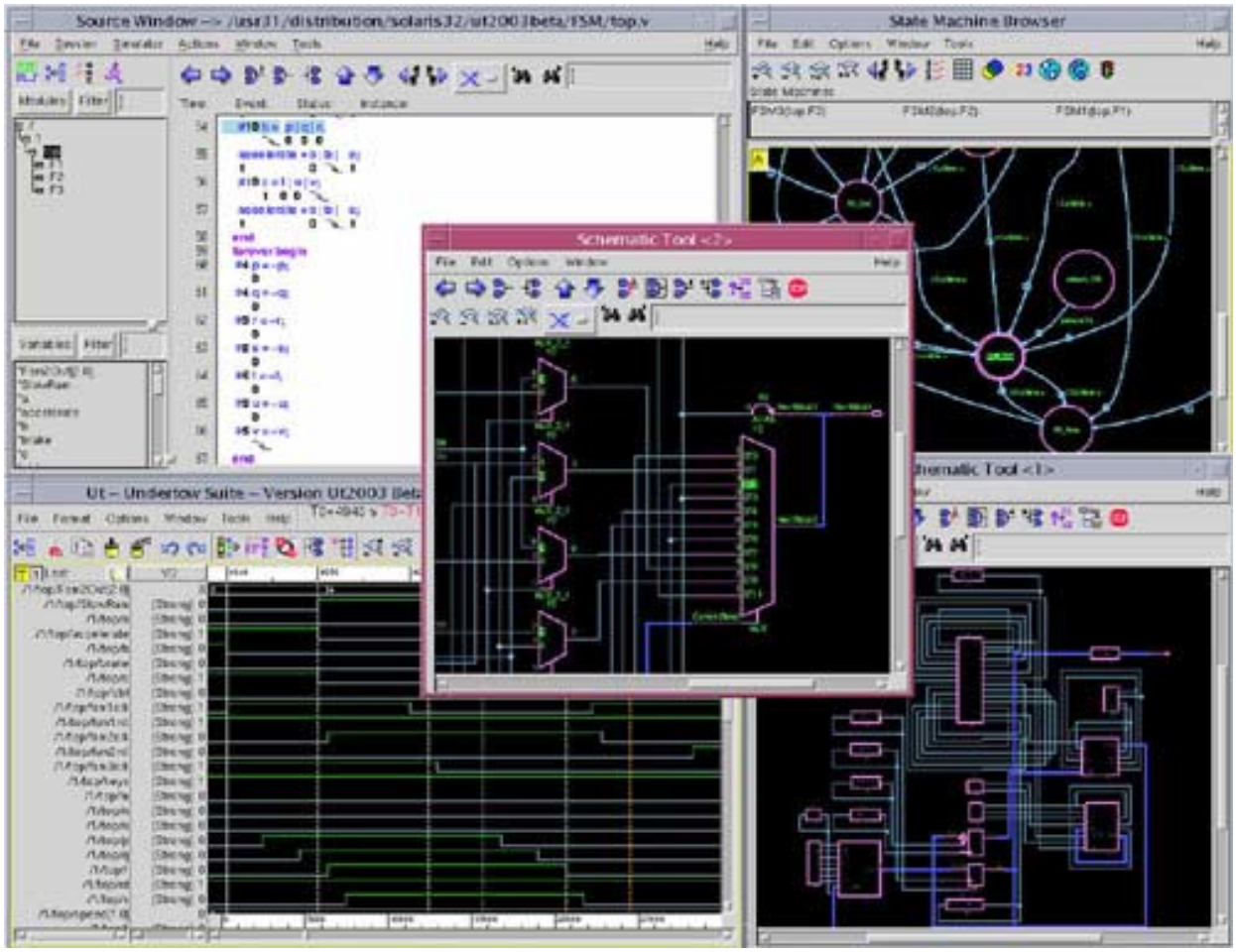
Undertow 2003

User Guide and Tutorial



Veritools

Version 2003.2.3
Version 2004.0.3
Jan 2005

Undertow 2003

User Guide and Tutorial

Veritools, Inc.

459 Hamilton Ave, Suite 200
Palo Alto, CA 94301

Phone: (650) 462-5590
Fax: (650) 462-5593

inquiry@veritools.com
www.veritools.com
www.veritools-usa.com
www.veritools-web.com

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

About This Manual

0.1 Copyright

All rights reserved. No part of this manual may be reproduced in any form or by any means without written permission from:

Veritools, Inc
489 Hamilton Ave. Suite 200
Palo Alto, CA 94301 U.S.A

Copyright ©1996-2003 Veritools, Inc.

0.2 Overview

Veritools is committed to its customers and will continue to deliver the industry's leading verification tools. Veritools' Undertow Suite is the latest upgrade to Undertow, the automated debugging tool used throughout the industry to enhance productivity for designers and testers world-wide.

0.3 Contents

This manual is organized into chapter by subject and levels of detail. The introduction is a summary of the tools in Undertow Suite, the user guides explain the basics for using to the tools, while the reference chapters give detailed explanations. The quick start guides explain the basics for using the tools for specific projects, Verilog, VHDL, and mixed mode. There are also chapters for PLI, Perl ,and third-party tools.

0.3.1 What's New for Undertow Suite 2003

Summarizes the new features included with latest Undertow Suite 2003 releases.

0.3.2 Introduction

Summarizes the features included with Undertow Suite

0.3.3 Software Installation and Setup

Describes how to download and install Veritools Undertow and Undertow Suite software products.

0.3.4 Undertow User's Guide

Describes the basic operations that you can perform using Veritools Undertow. This section introduces the main windows, describes how to start Undertow, and presents examples on how to use the product for common tasks.

0.3.5 Undertow Suite User's Guide

Describes the basic operations of the Undertow Suite. You can use the set of tools to debug source code for Verilog, VHDL, Verilog/VHDL mixed mode, and in future versions, SystemC. This section introduces the main windows, describes how to start the Undertow Suite, and presents examples on how to use the main windows.

0.3.6 Perl Scripting

Use the scripting tool to write scripts for common tasks, and then use the Builder tool to build a graphical window that facilitates the execution of this script.

0.3.7 Working with PLI

Undertows PLI (program language interface) enables the designers to go outside their Verilog design to another languages to create specific functions.

0.3.8 Verilog Quick Start Guide

This quick start guide uses a Verilog code sample to walk you through a demonstration of the basic functions of Undertow Suite.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

0.3.9 Undertow Integration Guide With Third-Party Tools

Discusses the Undertow-DFII Integration Kit, which integrates the Undertow Waveform Window into Cadence dfII.

0.4 Typographical Conventions Used

- Proper names for icons and windows are capitalized.
- Data to be entered as command line text are displayed in bold.

  setenv UT_ROOT_DI*R* <*full_path_directory*>

- Menu choices are displayed in italics, for example:

  Tools =>Trace

- Code or system information is displayed in the following form:

```
initial begin
  $vtDumpnames("fsm.sigs");
  $vtDumpvars();   /*dumps everything*/
 // $utConnectivity;
 #30000 $finish;
end
```

0.5 Technical Assistance

If you experience a problem installing or using Veritools products, contact your system administrator for system issues or Veritools customer support for product issues.

Veritools, Inc. Technical Support

Tel: 650-462-5590
E-mail: support@veritools.com

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Table of Contents

CHAPTER 1 Introduction 11

Overview **11**
Undertow Suite Functions **11**
Waveform **11**
Source Code **12**
Source Code Trace **12**
State Diagrams **13**
Gate and RTL Schematic **13**
PLI **14**
Perl Scripting **14**

CHAPTER 2 Software Installation and Set up 15

Description **15**
Supported Platform **15**
Supported File Format **16**
Obtaining Undertow and Undertow Suite Executable **17**
Obtaining Software Licenses **17**
Installing Undertow Suite on UNIX/Linux **18**
Installing Undertow/ Undertow Suite **18**
Installing Licenses on Unix/Linux **19**
Environment Variables in a Unix Environment **21**
Troubleshooting the License Manager in a Unix Environment **23**
Installing Undertow Suite in Windows **25**
How to setup the license in Windows **26**
Troubleshooting for License Manager in Windows Platform **29**

CHAPTER 3 Undertow User's Guide 31

Overview **31**
Starting Undertow **32**
Command Line Options **33**

 Copyright Veritools 2004 All Rights Reserved

Loading Files From the Menus 34
Working With Multiple Files 36
Viewing Waveforms 37
Module Hierarchy 37
Viewing Signals 38
Zoom Operations 40
Using the Mouse to Zoom 40
Panning 41
Setting Waveform Height 41
Placing Time Cursors 42
Placing Tags 43
Finding Edges And Values Of A Signal 44
Adding Panes 45
Bundling Signals 46
Working With Aliases 48

CHAPTER 4 **Undertow Reference Manual 51**

Description 51
Menu Bar 52
Cursor Time Values 53
Time Markers 53
Icon Bar 53
Signal Name List 53
Value List 54
Wave Display 55
Time Cursors 55
Time Scales 55
Scroll Bars 55
Main Toolbar Icons 56
Grouping Signals 63
Chooser Tool Bar Icons 66
File Menu 68
Open Additional File 68
Closing Files 70
Reread Dumpfiles 70
Saving Undertow Sessions 71
Restore 71
Start New Undertow 72
Sync 72
Printing Signals 73
Creating Files 76

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Format Menu **76**
Options Menu **78**
Text (Choose) **78**
Explode Bus **80**
Bundle Signals **81**
Alias By Name **84**
Alias By Value **86**
Pane Controller **90**
Properties **91**
Getting Information **93**
Recall Output **95**
Window Menu **96**
Configure Dialog **96**
Navigator **107**
Cursors **108**
Annotate Waveform **110**
Tags **110**
Fit To Pane **113**
Horizontal Scrolling **115**
Vertical Scrolling **117**
Set Display Interval **117**
Auto Update **118**
Real Time **118**
Time Scale **120**
Waveform Heights **122**
Sync Panes **122**
Adding and Removing Value Area **123**
Floating Windows **124**
Tools Menu **125**
Test Analyzer **125**
Connectivity Analyzer **127**
Logic Analyzer **128**
Toggle Analyzer **131**
Sequence Analyzer **132**
Analog Analyzer **132**
Application Note - Creating A Fast File **135**
Description **135**
Creating A Fast File from the User Interface **135**
Using The Vdump Routine **137**
Generating A Dump File (VCD) From Verilog **138**
Generating A Fast Dump File From Verilog **139**
Application Note - Analog Features **140**
Description **140**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Main Toolbar Analog Icons **141**
Chooser Tool Bar Icons **146**
Analog Configure Dialog **147**
Cursors **158**
Examples **159**
Zooming **159**
Window Sizing **160**
Pane Zoom Analog **160**
Measures **161**
Calculate Slope **162**
Overlay Waveforms **163**
Trigger Functions **164**
Analog Tools **166**
Command Line **183**
Functions Calculator **186**
Analog Filter Builder Guide **192**
Script Tool **196**
Plotting Functionality **214**
Graphics Capabilities **216**
User Interface Functionality **217**
Analog waveform processing capabilities **220**
FFT **221**
FFT Interface **221**
FFT Window Menu **222**
..... **223**
Retrieving the Data Point **223**
Saving Signals **224**
Restore Signals **224**
Example **225**
Simulation results comparison and analysis **228**
Compare transition times, from digital simulation result files **228**
Compare signal levels of digital simulation result files **228**
Comparing results of simulations **228**
Measuring time differences between signal transitions, on different signals **228**
Compare analog waveforms **228**
Absolute Value Function **229**
Compare crossing points over a user-selectable threshold **229**
Compare analog waveforms to digital waveforms **229**
Convert analog waveform to digital waveform **229**
Search for a point of interest on a curve so that it can be marked **229**
Output file compatibility **229**
Encapsulated PostScript (EPS) graph output, color or black-and-white **229**
PostScript (PS) file output, color or black-and-white **230**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

x-y value pairs **230**
Save results of arbitrary expressions, for reload and comparison **230**

CHAPTER 5 Undertow Suite User's Guide 231

Undertow Suite Overview **231**
Defining Mouse Controls **232**
Source Window **233**
Waveform Window **239**
State Diagram Window **242**
Schematic Window **244**
Starting Undertow Suite **246**
Command Line Options **247**
Loading Files From Menus **248**
Batch Operation of Undertow Suite **248**
Using a Command Line to Start the Undertow Suite **252**
Using Undertow Suite **256**
Stepping Through Source Code **256**
Displaying Loads and Drivers **257**
Using Breakpoints in the Source Window **258**
Execution Information in the Source Window **259**
Using the Waveform Viewer **260**
Displaying waveforms **260**
Snapping To The Exact Line of Source Code **261**
Using The Schematic Viewer **261**
Viewing Modules In Schematic Viewer **262**
Finding Connections **263**
Showing Multiple Scopes **264**
HDL Analyzer **264**
Trace Back **265**
Using The State Diagram Viewer **266**
Adding Panes to the State Window **267**
State Stepping (Forward / Backward) **267**
"How To" Procedures **268**
How To Select Source Code Files **268**
How To Show/Locate Source Code For A Module **268**
How To Find Signal Assignments/Use In Source Code **269**
How To Display a Signal From the Source Code Window **270**
How To Find Source Code From the Waveform Window **270**
How To Start the Schematic Window **271**
How To Trace A Signal In Schematic Window **272**
How To Highlight A Net In Schematic Window **273**
How To Locate a Schematic Element in a Source Code Element **273**

How To Display A Signal From the Schematic Window **273**
How To Find the Source Code From the Schematic Window **273**
How To Start the State Window **273**
How To Select a State Diagram **274**
How To Step the State Diagram **274**
How To Find Source Code From the State Window **275**
How To Show All Exercised States **275**
How To Show All Coincident States **275**
How To Display a State Table **275**
How to View Memory Contents **275**
How To Overlay Signals **276**
How To configure the Xsun server on Solaris to use a default 24-bit TrueColor colormap **277**

CHAPTER 6 Undertow Suite Reference Manual 279

Undertow Suite Overview **279**
Source Code Window **280**
Source Code Browser and Signal Selector **280**
Waveform Window **280**
State Diagram Window **280**
Schematic Window **281**
Undertow Suite Basics **282**
Starting Undertow Suite **282**
Interactive Simulation with Command Line **284**
Using GUI to start Undertow Suite **286**
Running Interactive Simulations **290**
Loading Files **294**
Source Code Window **294**
Source Code Window Icons **295**
Source Code Menus **298**
File menu for the Main Undertow Suite Window **299**
File Open Menu **300**
Description of the Sessions Menu **303**
Simulator Menu **310**
Actions Menu **311**
Window Menu **313**
Tools Menu **323**
Mouse Actions and Popup Windows **323**
“How to” Operations for Source Code Window **324**
Selecting source code files **324**
Showing and Locating source code for a module **325**
Finding a signal assignments/use in source code **325**

 Copyright Veritools 2004 All Rights Reserved

Displaying a signal from source code window **326**
Finding source code from the signal window **326**
Schematic Window **327**
Description **327**
Starting the Schematic Browser Tool. **328**
Toolbar of Schematic Browser **329**
Schematic Toolbar Icons **329**
Schematic Menus **331**
File Menu **332**
Edit Menu **335**
Options Menu **335**
Window Menu **337**
Trace Back **341**
“How to” Operations, for the Schematic Browser **341**
Tracing a Signal in the Schematic Window **341**
Highlighting a Net in the Schematic Browser **342**
Locating a Schematic Element in a Source Code Element **342**
Displaying a Signal from the Schematic Window **342**
Finding the Source Code from the Schematic Window **342**
State Machine Browser **343**
Description **343**
Starting the State Tool **344**
State Toolbar **344**
State Menus **345**
File Menu **346**
Edit Menu **347**
Options Menu **348**
Window Menu **349**
“How to” Operations, for State Machine Browser **349**
Loading a Module **349**
Multiple Views **349**
Setting the Active View **350**
Stepping the State Diagram **350**
Stepping the State Browser **351**
Finding Source Code from the State Window **352**
Showing Persistence **352**
Analyzing Coincidence **352**
Trace Window **353**
Application Notes **356**
Integration with Gate level netlist **356**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 7 Perl Scripting 359

Script Tool **359**
To Download Undertow Perl Package **359**
To Install Perl package **359**
Script Tool **360**
Perl Configuration **361**
Overview of the Perl Language **362**
Operators **364**
Functions **366**
Example Script **371**
Builder, a graphical Perl window building tool **373**
Widgets **374**
Undertow Analog Extensions **390**
Downloading Scripts **390**
Extracting Scripts **391**
Scripts using Perl Analog Extension **391**
UndertwoAnalogExtension.pm 392
UndertowPerlUtilities.pm 397

CHAPTER 8 Working With PLI 481

Overview **481**
Optimizing Tool **481**
Starting the Optimizing Tool **482**
$VtDumpvars Parameters **482**
Compression Modes **483**
Dump File Names **484**
Turning Dump File on/off **485**
Trace Files **485**
Creating A Trace Dump File, Using Cadence Verilog-XL **485**
Creating A Trace Dump File, Using VCS and FinSim **486**
PLI Dumping **487**
General Descriptions **487**
Connection Analyzer **488**
Use the $utConnectivity system task **488**
Use the Iv parser if you want to create an RTL schematic **488**
Creating a Memory Dump File (mem.dump) **489**
How to View Memory Contents **489**
Creating A Fast File On The Fly ($vtDump) **490**
Limiting A VCD File **490**
Linking In PLI Routines **491**
Downloading the FSM Example from the Veritools Web Site **491**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Synopsys VCS **491**
Cadence Verilog-XL **497**
Cadence NC Verilog **499**
Modeltech Vsim **504**
Fintronic FinSim **507**

CHAPTER 9 Quick Start Guide: Verilog Example 401

Preparing for Verilog Quick Start **401**
Defining Mouse Controls **401**
Setting Environment Variables **402**
Starting Verilog Quick Start **402**
Entering Start Command **402**
Viewing Undertow Suite **404**
Using the Basic Features of Undertow Suite **406**
Using the Hierarchical Browser **407**
Displaying Selected Signals in the Waveform Window **408**
Displaying Source Code for any module **409**
Finding a specific Variable **410**
Virtual Simulation Feature **410**
Snap to Source **411**
Tracing Back the Drivers to Their Ultimate Source **414**
Using the Advanced Features of Undertow Suite **416**
Displaying Specific Windows **416**
Schematic Tutorial **417**
Highlight Net **429**
Application Notes **435**
Integration with Gate level netlist **436**
State Diagram Window **439**
Overall Demonstration **442**
Interactive Mode **446**
With No Trace Enabled **446**
Trace Enabled **450**
Linking PLI Into Your Verilog **453**
Linking in the PLI and Building a Verilog Executable, Verilog-XL **453**
Using Other PLI Routines **456**
Installing Veritools PLI Routines for VCS **456**
Installing Veritools PLI Routines for Modeltech **462**
Adding the Veritools FLI to MTI 5.3+ (Solaris, HP-UX, Linux) **464**
Installing Veritools PLI Routines for NC Verilog **468**
Installing Veritools PLI Routines for Finsim **474**

 Copyright Veritools 2004 All Rights Reserved

CHAPTER 11 Undertow Integration Guide With Third-Party Tools 515

Introduction **515**
Undertow-DFII Integration Kit **515**
Installation **516**
Environment Setup **516**
Standalone Usage Of Undertow-Cadence DFII Integration **517**
Usage **517**
Instance Hierarchy Traversal **529**
Using Undertow in the Cadence DFII Verilog Integration Environment **530**
Usages **530**
Signal Probing In Verilog Integration Environment **532**
Preparing a Verilog VCD File For Undertow **533**
Preparing Veritools dump file for Verilog-XL **533**
Preparation Of Verilog Netlists In Verilog Integration Environment **534**
Preparing Netlists In Antrim Integration Environment **534**
Interface With Undertow From Other Applications **535**
Invoking Undertow From Other C Applications: **535**
Start Undertow with command line arguments: **535**
Set up a link between Undertow and Application by a C routine: **535**
Current Supported Undertow Commands **537**
ADD <options> <signal_name> [<alias_signal_name> **537**
ADD-ALL <options> module_name **537**
ADD-TO-OVERLAY <-color color_name> signal_name **538**
BUNDLE <options> bundle_name <signal_name> <signal_name> **538**
DELETE <signal_name> **538**
GET-COLOR <signal_name> **539**
<options> **539**

Index **1**

 Copyright Veritools 2004 All Rights Reserved

CHAPTER 2

Introduction

Overview

Welcome to Veritools' Undertow Suite, the complete and only universal world-wide popular design analysis work bench. Undertow Suite significantly improves designer productivity and accuracy through a flexible, intuitive analysis environment for design debugging.

Undertow Suite Functions

Undertow Suite provides a variety of powerful, easy to use tools for the digital and analog markets.

Waveform

The Waveform Window instantly displays waveforms from very large waveform files. View any number of files, both analog and digital, within an unlimited number of waveform panes for comparisons.

Digital and Analog

- Color code, group, and arrange signals to make the information stand out visually in the windows
- Waveform search function for locating wave transitions or user-defined values.
- Users can run the Test Analyzer to compare waveforms and display the differences.

- Use the three cursors, T0, T1, and T2 while analyzing the waveforms.

Analog Only

The Analog Analyzer provides users with a complete and powerful analog analysis capability, included are waveform overlay capability and complex operators.

Source Code

Users can instantly go to the exact line of source code from any signal transition on the waveform window.

- Double clicking on any signal will display where this signal has been assigned.
- Users can double click on any task or function and instantly see the code that is part of this task or function.
- Step the source code forward or backwards in the Source Code Window.

Source Code Trace

Undertow Suite allows users to trace their exact source code execution, and all features work with trace on or off.

- From the Trace Window users can step the simulation both forward or backward.
- Trace operations can be restricted to module, number of modules, or to a time period.
- Users can use the Trace Window in batch mode, allowing them to step the simulator forward and backward without tying up their simulator license.
- Drag and drop to display signals, modules, and variables between the Trace Window and other windows
- Users can trace signal through multiple modules and signal name changes.
- Trace back allows a user to double click on any variable and have the Source Code Window display where this variable is assigned. When there is no trace file, this interactive tool will use the parsed Verilog source files.
- Users can snap to any transition of a signal in the Waveform Window and instantly display the exact line of source code that caused this transition on the Source Code Window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- When users are looks in at a an assigned signal (left side of expression) that is transitioning, they can quickly see exactly what signals on the right side of this expression caused this transition. By double clicking on these signals, users can back trace to any point to find the original reason for a signal transition.
- Snap to source

State Diagrams

State diagrams are automatically generated from the user's source code. Detailed state analysis is provided to allow users to see exactly how different parts of their designs react with one another, and determine what additional tests are required to completely verify a design.

- The State Diagram Window provides the user with several powerful new analysis features including: forward and backward state stepping, state path analysis, state coverage and state coincidence analysis.
- Users can display a single state diagrams or multiple state diagrams to see how all of these different state diagrams sequence together.
- Users can see how state machines interact within the same time point.
- Users can select a single state on any diagram and fully explore what other state machines are doing during this state.

Gate and RTL Schematic

RTL and gate schematics are automatically generated directly from the user's source code files.

- RTL-level schematics allow the user to see all sections of their RTL design graphically and to quickly find drivers and loads from the Schematic Window.
- Both debugging and comprehensive schematic generation are provided for both RTL and structural Verilog design.
- Users can go backward to find any driver or forward to locate any load on the schematic. Even large schematic files can be loaded and displayed quickly
- A powerful comparison feature is provided which provides for example RTL to gate simulation comparisons, which can then be linked to schematics or source code to identify differences.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

PLI

Undertows PLI (program language interface) enables the designers to go outside their Verilog design to another languages to create specific functions.

- VeriPower in an additional Veritools product, it is a PLI based power calculation tool that produces power calculations targeting user who want fast but very accurate look at the power consumed by their designs while running a Verilog simulation.
- Power Tool is an additional Veritools product, it is a PLI based power calculation tool for Verilog designers, targeted at users who want a quick an convenient look a the power consumed by their designs while running a Verilog simulation.
- Toggle Tools is an additional Veritools product, it is a PLI routine that performs a toggle coverage analysis of the Verilog design during the simulation.

Perl Scripting

Undertow includes a completely integrated Perl scripting tool that can be used for digital or analog analysis.

- Scripts can be run interactively or in batch mode using the Veritools databases.
- This scripting tool can be use to do repetitive operations automatically, and can even build complex windows quickly.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 3

Software Installation and Set up

Description

This chapter describes how to download and install Veritools Undertow and Undertow Suite software products. Undertow is a powerful waveform display tool for viewing digital, mixed mode, and analog simulation output form a variety of simulators, including:

- Verilog
- TIMEMILL
- POWERMILL
- HSpice
- VHDL (VSS, LEAPFROG, MENTOR, and ModelTech)
- Anagram

Supported Platform

Undertow supports the following Platform:

1. Linux
2. 32/64-bit Unix (Solaris 2.7 or later, SOL 7.0, SOL 9.0 +; HP-UX 9.0)
3. Windows (NT, 2000, XP)
4. IBM AIX 4.3

Supported File Format

TABLE 1. Support File Format

Analog Format	Digital Format
Hspice	**VCD**
Hspice raw files	VCD Extended
TR0	Mentor .wlf
AC Sweep	PLI 1 and PLI 2, VPI
DC Sweep	Compression of 650 to 900 times
FT0	Fast formats, almost any file compressed
CSDF	by 650-900 times via vdump
TR0	FLI
ACSweep	VHDL PLI, VPLI
DC Sweep	Veritools API, optimized format,
FT0	Compression of up to 900 times using the Veri-tools API
Pspice	WGL test format
Nutmeg Format	
Nanosim	
.out format	
Native Format via the API	
Powermill/Timemill/Railmill	
.out	
Mentor Eldo	
.cou, .dou	
Smartspice	
raw files	
Sweeps	
IBM	
GRF1 format	
Starsim	
Output fast files	
Specture	
WSF	
PSF	

This includes virtually all analog and digital simulators including VHDL, Verilog, and SystemC. Veritools' customer support includes feature requests and upgrades, as well as requests for adding new formats. We diligently support our customer's needs at all times.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Obtaining Undertow and Undertow Suite Executable

You can obtain Undertow via the web. You must then receive a product or demo license for a Veritools representative to use the Undertow Suite. See “Obtaining Undertow and Undertow Suite Executable” on page 22.

- Contact us to request for a copy
- For web access, point your browser to *www.veritools-web.com* and choose the **Download** tab near the top of the screen. From the various lists and sublists you can choose the appropriate version of Undertow Suite.

  You will be able to download the executable and the manual

Obtaining Software Licenses

Undertow requires each user to have a valid license. To obtain either a demo license or a permanent license, contact your Veritools sales representative, or send a request to:

Office: (650) 462 - 5590
Fax: (650) 462 - 5593
E-mail: inquiry@veritools.com

The information you should provide to Veritools. when applying for a license includes the following:

- Name of the software product and number of copies desired.
- Is this license for evaluation copies or purchase copies? If you are ordering purchase copies, please provide purchase order numbers
- Host information:
  - Hostid (number) on SUN systems is accessed by typing *hostid* or *lmhostid.*

    Note: ***lmhostid*** *is a program that comes with the license manager software.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- For a hostid (number) on Hewlett Packard (HP) systems you must first login as *su* (superuser to "root") then get the host id. Use */etc/lanscan* to determine the hostid based on ethernet address
- For Linux systems use */sbin/ifconfig etho 0*, the host id is the ethernet address without the colons.
- For Window systems the choices depend on your hardware and software setup, so type:
  - *dir c:* /* Use the Volume Serial Number without the dash */
  - *lmutil lmhostid* or *lmhostid -flexid* or *lmhostid -cpu* or *lmhostid -cpu96*

    Note: Verify with the license manager software, which method is most appropriate to use on your system.

- Hostname (name), on all systems type *hostname*.
- Fax number.
- E-mail address for contact.

A Veritools representative can give you access to the User manual, as well as offer assistance with downloading or license installation.

Installing Undertow Suite on UNIX/Linux

Installing Undertow/ Undertow Suite

To install on Unix/Linux

1. Make sure you have a enough disk space to unzip and install the file.

   Note: When unzipped, Undertow Suite requires about 15 Megabytes of hard disk space.

2. After you complete the download described in "Obtaining Undertow and Undertow Suite Executable" on page 22, move the compressed Undertow package to the desired directory.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- If the package is in a gzip format (gz suffix), then to decompress the file, use the *gunzip* utility:

  gunzip ut*.tar.gz**

- If the package is of a standard UNIX compression (Z suffix), then to decompress the package, apply the *uncompress* command to it:

  uncompress ut*.tar.z**

3. After you decompress the Undertow distribution, use the *tar* command to untar the distribution:

   tar -xvf ut*.tar**

   For more information about these commands, consult the UNIX *man* pages for *uncompress*, *gzip*, and *tar*.

   After you installing the software you must configure the software licenses before running Undertow Suite.

Installing Licenses on Unix/Linux

Veritools uses the FLEXlm license server to assign and control Undertow software licenses. If you are using other FLEXlm licensed products, then you can add the Undertow license keys to your existing FLEXlm license file. The Undertow distribution contains all of the necessary FLEXlm license software tools (such as **lmgrd** and **lmstat**) to run Undertow.

When you receive your Undertow license, you need to place it in either a new file, or an existing FLEXlm license data file. Undertow accesses this license file via one of two possible environment variables, **LM_LICENSE_FILE**, or **VERITOOLS_LICENSE_FILE**.

Note: The Veritools demo license does not require you to install either FLEXlm, or the lmgrd file that FLEXlm uses.

Setting up a Demo License in Unix

1. After you obtain the demo license(s), you must copy them into a plain text (ASCII) file. You can use any text editor, such as *vi* or *Emacs*, to create this file.

   *Note: You must place each feature on a new line. Each feature must be contained **only** on one line (only one carriage return, located at the end of each FEATURE line). Below is an example of a file called "license.dat"; your license file should look similar to this example.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FEATURE undertow verid 8.000 31-dec-2000 20 4C264F3457312DF11234344 DEMO "DEMO"
FEATURE optimizing_tool verid 8.000 31-dec-2000 20 54567DE34555F8E834 DEMO "DEMO"

2. Use the UNIX **chmod** command to set the permissions for your license file—for example:

   chmod 444 license.dat

   *Note: The license file (**license.dat** in this example) must be readable for all authorized Undertow users.*

3. In your shell start up script (*.cshrc* for C Shell), set the LM_LICENSE_FILE environment variable to point to your license file—for example:

   setenv LM_LICENSE_FILE /usr/local/veritools/license.dat

Setting up a Permanent License in Unix

- If you already use Globetrotter FLEXlm to license other software products, then you can add all the **FEATURE** lines from your Veritools license file to your current license file.
- If you are not using the demo license, then add the **DAEMON** line from your Veritools license file to your license file. For example:

  DAEMON <*verid*> <*your_verid_executable_path*>

  where *verid* is the Veritools' daemon executable, located in the *your_verid_executable_path* directory.

- If do not want to modify your current license file, then assign a unique file name to your Veritools license file (such as **ut_licence.dat**), and set the **VERITOOLS_LICENSE_FILE** environment variable to point to this file.

For example, if you put the license file in the **testdir/ut_license.dat** file, then for a C Shell:

setenv VERITOOLS_LICENSE_FILE /testdir/ut_license.dat

or

setenv VERITOOLS_LICENSE_FILE 8182@elisa:1900@voy1

where **elisa** and **voy1** are the license servers, and **8182** and **1900** are the used port numbers defined from your license file.

 Copyright Veritools 2004 All Rights Reserved

Undertow software checks for the **VERITOOLS_LICENSE_FILE** environment variable or the **LM_LICENSE_FILE** variable. If you did not set either of these variables, Undertow looks for the default file.

/usr/local/FLEXlm/licenses/license.dat

1. To start the license manager and create a log file to append to, type:

   lmgrd -c *your_license_file >lmgrd.log 2>&1*

2. To check whether your license daemon is running, type:

   lmstat -a *your_license_file*

   If your license manager is running and the Veritools features are available, then the license manager should return the following to your terminal:

```
/FLEXlm/lmstat -a /FLEXlm/license.dat
lmstat-Copyright (C) 1989-1994 Globetrotter Software, Inc.
Flexible License Manager status on Tue 9/21/99 15:41

License server status (License file: 8182@elisa):
    elisa: license server UP (MASTER)
Vendor daemon status (on elisa):
    verid (v3.x): UP

Feature usage info:
Users of undertow: (Total of 20 licenses available)
Users of optimizing_tool: (Total of 20 licenses available)
Users of interactive_tool: (Total of 20 licenses available)
Users of behavioral_tool: (Total of 4 licenses available)
```

See “Obtaining Undertow and Undertow Suite Executable” on page 22.

Environment Variables in a Unix Environment

Shared Environment Variable Configuration

Undertow and Undertow Suite use several system environment variables: **UT_ROOT_DIR**, **UT_WORK_DIR**, **VERITOOLS_LICENSE_FILE**, **EDITOR**, and **PATH.**

- **UT_ROOT_DIR** points to the top level of the Undertow or Undertow Suite software installation. For Undertow and Undertow Suite to run properly, you must set this variable. To set this variable in a C Shell, **cd** to the Undertow Suite install directory, and type:

  setenv UT_ROOT_DIR *undertow_install_directory*

where *undertow_install_directory* is the full directory path name of the Undertow installation directory.

- **UT_WORK_DIR** sets the default directory for Undertow and Undertow Suite file dialogs**.** To set the working directory to the current directory in a C Shell, type:

  setenv UT_WORK_DIR .

  or

  setenv UT_WORK_DIR *working_directory/directory*

  where <*work_directory*> is the full directory path name of the Undertow user's working directory, and *directory* specifies the **name** of the user's working directory.

- **VERITOOLS_LICENSE_FILE** defines the License file for Veritools products. This variable overrides the **LM_LICENSE_FILE** environment variable.

  The VERITOOLS_LICENSE_FILE variable lets administrators set up Undertow licenses without changing current LM_LICENSE_FILE settings. The file specified in the VERITOOLS_LICENSE _FILE variable contains either demo keys or permanent keys. For more information about how to obtain and install Veritools licenses, refer to Section , "Obtaining Software Licenses".

- **EDITOR** selects which editor Undertow or Undertow Suite calls when you edit files. You can set this variable to any valid editor on your system, such as vi, Emacs, xemacs, jot, dtedit, and so on. For example in the C Shell use:

  setenv EDITOR vi

- **PATH** specifies a set of directory paths that the Unix operating system searches to locate executable programs. If you include the directory that contains the executable for Undertow, then you can start Undertow (Suite) without specifying a directory. That is, if the Undertow executable is in a directory path that you specified in the Unix PATH variable, then you can specify only the executable name when you start the software:

  ut

- **PERLLIB** specifies the path to the Perl libraries. Set this environment variable if you want to use a different Perl library (if you want to override the default Perl library setting).

Individual Environment Variable Configuration

Undertow and Undertow Suite use several system environment variables that may be configured for a group of users (See "Shared Environment Variable Configuration" on page 26.), or each user may customize their own copy.

 Copyright Veritools 2004 All Rights Reserved

1. Create a directory called app-defaults on you home drive.
2. Copy the Undertow executable file(UNdertow) stored in the $UT_ROOT_DIR and place it in the new app-defaults directory.
3. Set the environment variable XAPPLRESDIR to $home/app-defaults (C Shell):

   setenv XAPPLRESDIR <home-dir>/app-defaults /* Capitalized UN */

4. Use an editor to open the UNdertow file, such as the *vi* editor.
5. For each line where you want to alter the command parameters:
   - Remove the '!' (bang) character from the first space in the line to uncomment and enable that line.
   - Change the value for the parameter as desired.

Troubleshooting the License Manager in a Unix Environment

This is the troubleshooting guide for installing the license manager software when using Veritools's software products.

1. The server for the license manager and all remote workstations that will run the licensed software must have a license.dat file holding the information from the license file that has been sent to your site.

   Check your workstation to make sure that it has a license.date file. The license.dat file should either be located in the default location, /usr/local/flexlm/licenses/license.dat or in another location of your choice. If it is not in the default location then the path to license.dat must be specified in the environment variable: LM_LICENSE_FILE.

   To set this variable LM_LICENSE_FILE type:

   setenv LM_LICENSE_FILE (path)/license.dat

   On the server, if the license.dat file is not in the default path then when you start "lmgrd" (from /etc/rc.local) you must specify the path with the -c option:

   (path)/lmgrd -c (path)/license.dat > (path)/log 2>&1 &

   If you are starting "lmgrd" manually from the csh then make sure you specify the path using the -c option as show below:

   (path)/lmgrd -c (path)/license.dat >& (path)/log &

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Check that the path where you have put verid is the same that is specified in the license.dat.

   i.e. DAEMON verid /usr/local/flexlm/daemons/verid

3. If you modify the license.dat file while running “lmgrd” you should reinitialize “lmgrd” by running “lmreread”.

   lmreread

4. The license information on the FEATURE line should be inserted into the license.dat file exactly as it is sent to you. Please check this, including the FEATURE name, number of licenses, expiration date, and license key.

   i.e. FEATURE vltool verid 1.000 2-aug-92 10 AZBXCYDWEVFUGTGS “ “

5. After you have started “lmgrd” you can check that it is started by using “lmstat -a”. In addition, after it is started you should check to see that the daemon verid is running using ps (1) and grep (2)

   ps lmstat -a

   ps -ax | grep verid

   ps -ax | grep lmgrd

If verid is not running kill (1) “lmgrd” and restart “lmgrd” as shown as in step 1 above.

kill processid# /* processid# is returned by the ps command above */

If when you start “lmgrd” a message is returned that says “sock bind: address in use”

use ps(1) and grep(1) to see if there is another “lmgrd” running.

ps -aux | grep lmgrd

6. If there is another “lmgrd” running merge the DAEMON and FEATURE lines from the Veritools’ license.date into the already existing license.date file. The existing license.dat file will either be in the default path /usr/local/flexlm/licenses or it will be specified in the environmental variable LM_LICENSE_FILE.

   printenv LM_LICENSE_FILE

   If the socket is already in use due to a DIFFERENT program you need to use netstat(9) to select another socket number. The default socket address issued with the Veritools’ license is 1700. We suggest you choose 1701.

   netstat -an | grep 1701

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

If you receive only your prompt with this command, the socket address 1701 is available to use. If you receive anything else select another socket address and use netstat to verify that it is available. This number replaces the last number in the first line of the license file:

i.e. SERVER veritools 55486940c 1700 (1700 is the socket number)

Installing Undertow Suite in Windows

1. First, You need to download the latest file from the web-site E.g. ut2003.1.2.exe
2. Ensure that you have uninstalled ALL previous instances of undertow. To uninstall undertow, go to Start -> Settings -> Control Panel -> Add/Remove program. Select 'undertow' and press the 'Add/Remove' button. You should also reboot the PC
3. Install undertow by running the ut900_020328.exe. It is recommended to use the default directory as the target directory(which is c:\undertow)
4. At the end of undertow install, it will ask if you want to install the WinaXe X server.Unless you already have you own X server, you should answer Yes. Follow the winaXe installation. The attached version is a demo version of it. It runs for 30 minutes and then requires you to restart it. We will send you your permanent license for Winaxe together with your permanent license for undertow. The 'license' would be in a form of a new xwpdllid.dll file, which you would have to copy to the Winaxe program directory, which is usually at "c:\Program Files\LabF.com\WinaXe\"
5. Get the license file from Veritools and put it in the c:\flexlm\license.dat file.
6. Run lmgrd (the license manager daemon) by selecting: Start -> Programs -> Veritools -> lmgrd If your license is fixed node, or a demo license, this step is not required.
7. After installation is over. Run the X server by selecting: Start -> Programs -> WinaXe 6.2 -> Xsession.The first time you run Xsession it asks you if you want to be connected to the labf web site. Choose No. In the future you won't be needing to run X. Undertow will do that for you.
8. Run Undertow by selecting: Start -> Programs -> Veritools -> Undertow

How to setup the license in Windows

In general you don't need to run manually the lmgrd (the license manager), Under tow will try to start it automatically for you. However, there are three different configurations:

1. "Node locked" license
2. "Floating license" - the license server runs on the local computer
3. "Floating license" - the license server runs on a remote computer(unix or NT)

We at Veritools usually use the "floating license". Of the above 3 configurations, #1 and #3 don't require to run the lmgrd on the local computer.

NOTE: MAKE SURE THAT FOR UNDERTOW 9 YOU ARE USING THE LATEST VERID 7.2 ON THE SERVER THAT RUNS THE LICENSE.
To check the version number just *run verid.exe* and it should say:

(verid) FLEXlm version 7.2i
(verid) Vendor daemons must be run by lmgrd

The latest lmgrd and verid for both UNIX and NT systems are on the Veritools web site.

"Node locked" license

The license is good to run undertow only on the same computer where the license is located. You need to setup properly the environment variable LM_LICENSE_FILE to point to the location of the license file. E.g:

"LM_LICENSE_FILE=c:\flexlm\license.dat"

The license file license.dat might look like this:

FEATURE undertow_nt verid 9.000 10-may-2004 uncounted HOSTID=2343FA98

SIGN=93BB7344865E

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Or it might look like this:

SERVER my_nt_computer 3466716e 5490

DAEMON verid c:\flexlm\verid.exe

FEATURE undertow_nt verid 9.000 10-may-2004 uncounted

HOSTID=2343FA98 SIGN=93BB7344865E

"Floating license" - the license server runs on the local computer

The license is good to run undertow on any computer - on the current one, andon any other, unix or windows. You need to setup properly the environment variable LM_LICENSE_FILE to point to the location of the license file. And you NEED to run the lmgrd.exe program. (Start -> Program --> Veritools --> Lmgrd) E.g:

"LM_LICENSE_FILE=c:\flexlm\license.dat"

The license file license.dat for 5 licenses might look like:

SERVER ultrabook 80cc016e 5490

DAEMON verid c:\flexlm\verid.exe

FEATURE undertow_nt verid 9.000 10-may-2004 5 SIGN=93BB7344865E

"Floating license" - the license server runs on a remote computer (unix or NT)

The license is good to run undertow on any computer - unix or windows. You need to setup properly the environment variable LM_LICENSE_FILE on your local computer to point to the port and name of the remote computer where the lmgrdis running.

E.g:

"LM_LICENSE_FILE=5490@ultrabook"

Pay attention - The name of the computer that runs the license manager has to be known to the local computer. To verify that type:

ping <host_name> (e.g. ping ultrabook)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

If the local computer doesn't recognize it, you need to find its IP address, and update the \winnt\system32\drivers\etc\hosts file adding an extra line. E.g. After the update the hosts file should like like:

```
# Copyright (c) 1993-1995 Microsoft Corp.
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows NT.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
# 102.54.94.97 rhino.acme.com # source server
# 38.25.63.10 x.acme.com # x client host

127.0.0.1 localhost

10.1.1.123 ultrabook
```

Note - if you cannot update the host file, you can just use the host ip number. This might happen if you get the following error message:

lmgrd is not running: Cannot find SERVER hostname in network database

(- 14,7:11001 "WinSock: Host not found (HOST_NOT_FOUND)")

It means that there is a problem with resolving the 4 numbers ip out of the hostname. This can happen if you use the ip address on the local computer

(e.g. LM_LIENSE_FILE=5490@10.1.1.123)

and the license file on the remote computer uses the computer name instead of the host ip number

(e.g. SERVER ultrabook 80cc016d 5490).

To resolve it change the license file and use the ip address instead of the computer name.

(e.g. SERVER 10.1.1.123 80cc016d 5490)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Make sure that the port@host which appears on the environment variable setup of LM_LICENSE_FILE on your local machine has the exact same ip address and port number as the license file on the remote server.

For example - if you set your environment variables as follows:

LM_LIENSE_FILE=5490@10.1.1.123

And the remote license looks like this:

SERVER ultrabook 80cc016e 5490

DAEMON verid c:\flexlm\verid.exe

FEATURE undertow_nt verid 9.000 10-may-2004 5 SIGN=93BB7344865E

Then change this license to the following format:

SERVER 10.1.1.123 80cc016e 5490

DAEMON verid c:\flexlm\verid.exe

FEATURE undertow_nt verid 9.000 10-may-2004 5 SIGN=93BB7344865E

Don't forget to shut down the lmgrd and run it again after every change in the license file. You can check the verid.log file to see the output of the license manager.

To check if the license is running type:

lmstat -a

To stop the license daemon use:

lmutil lmdown

To start the license daemon use:

lmgrd

Troubleshooting for License Manager in Windows Platform

1. Undertow-NT works on windows platform, on NT, windows-2000 and windows-XP. There are some known issues running it on windows-98 or lower.2. No communication to the X server -> Verify that the environment variables are set correctly. They should be :

   UT_WORK_DIR=c:\undertow

UT_ROOT_DIR=c:\undertow

DISPLAY=127.0.0.1:0.0

LM_LICENSE_FILE=c:\flexlm\license.dat

LM_LICENSE_PATH=c:\flexlm:

2. You can check them by running the command prompt, and typing: 'set'. On windows-2000 to change the variables use Start -> Settings -> Command Panel -> System -> Advanced -> Environment variables. Pay attention that there is no double definition of the DISPLAY parameters in the User variables and the System variables.

3. If undertow doesn't run, and crash immediately without showing any information, try running undertow directly from the command line window. Go to the directory:c:\undertow and type ut.exe and see what the issues are.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Overview

This chapter describes the basic operations that you can perform using Veritools Undertow. Undertow is a powerful waveform display tool for viewing digital, mixed mode, and analog simulation output form a variety of simulators, including:

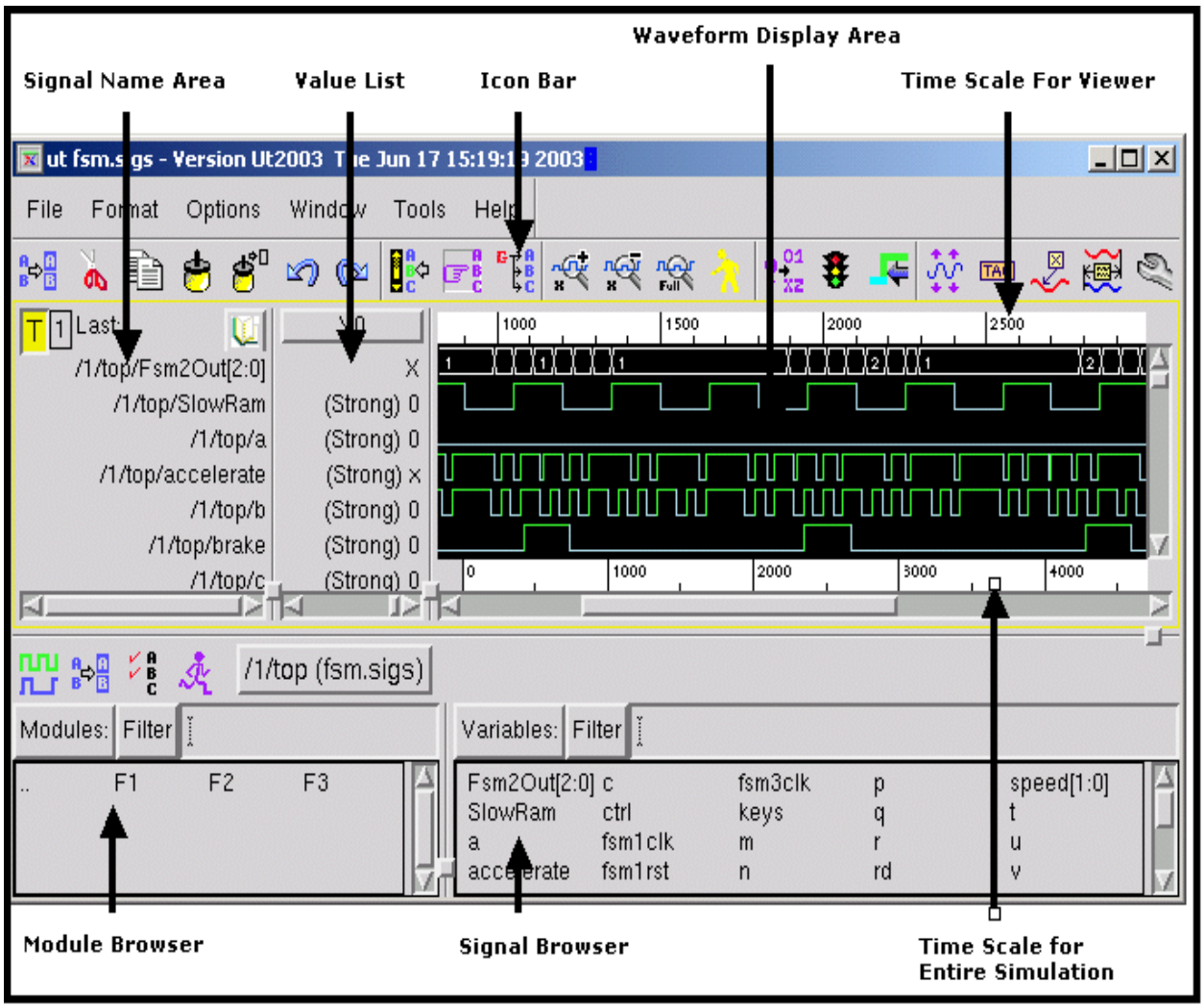
- Verilog
- TIMEMILL
- POWERMILL
- HSpice
- VHDL (VSS, LEAPFROG, MENTOR, and ModelTech)
- Anagram

Undertow lets you:

- interactively view output files, in batch or real-time
- trace netlist connectivity
- compare files
- perform toggle analysis

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 1. Undertow Main Window



Starting Undertow

To start the Undertow software:

1. Make sure that the directory containing the **ut** executable is included in your Unix path definition.

2. Make sure that all environment variables are correctly set, including:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- UT_ROOT_DIR
- LM_LICENSE_FILE or VERITOOLS_LICENSE_FILE
- UT_WORK_DIR

Command Line Options

To load files, you can select from Undertow's menu options as described in Section , "Loading Files From the Menus", or you can use the command line. Using command line options lets you quickly start and load files from a Bourne or C shell script. Table 2, "Undertow Command Line Options" lists some of the most common command line options used with Undertow.

TABLE 2. Undertow Command Line Options

Option	Description
-iv	Starts Undertow Interactive Suite.
-batch	Runs Perl scripts on files, without starting the GUI.
-v <*dumpfilename*,..n>	Loads the specified dump file(s), and bypasses the initial file dialog box.
-vf <*list_of_files_file*>	Loads multiple files from the list in a file.
-c	Uses a private color map. This option is useful if your system has low video RAM, or runs other applications at the same time as Undertow (such as Netscape) that take up most of the color resources. Use this option if he Undertow window opens without colors.
-/ *scope_separator*	Sets the scope separator for hierarchical names.
-com *command_script_file*	Runs the specified command script file.
-perl *script*	Runs the specified Perl script file when you open the GUI.
-r	Realtime mode. Loads a dump file that simulation is still creating. The default update time is 20 seconds.
-t	Time interval. Use this option to change the default update time for realtime mode.
-h	Lists all of the available command line options.
-s <*savefile*>	Loads the specified save file. A save file contains attributes and settings from a previous Undertow session.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 2. Undertow Command Line Options (Continued)

Option	Description
-S *realtime_savefile*	Restores the specified *realtime_savefile* whenever it changes.
-map <map_file_name>	This option allows mapping between different hierarchical names in the source code parsing and dump files. The map file is a simple ASCII file with one hierarchical mapping per line. A line has the format : source.hierarchical.name = dump.file.hierarchical.name When mapping is performed, only the first matching map name will be used, so the file should be start with the most specific mappings and end with the most general mappings.

The following are examples of using command line options:

ut -iv -prj FSM/smallfsm.prj -map FSM/smallfsm.map
ut -v fsm.dump -s fsm.save
ut -c -v fsm.dump
ut -vf xyz.file

All of these commands display the Undertow window (Figure 1, "Undertow Main Window").

Loading Files From the Menus

If you type **ut** without the **-v** option to start Undertow, two windows appear:

- The Undertow window itself (Figure 1, "Undertow Main Window").
- A file dialog box (Figure 2, "Open File Dialog"), which asks you to enter the name of the input file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

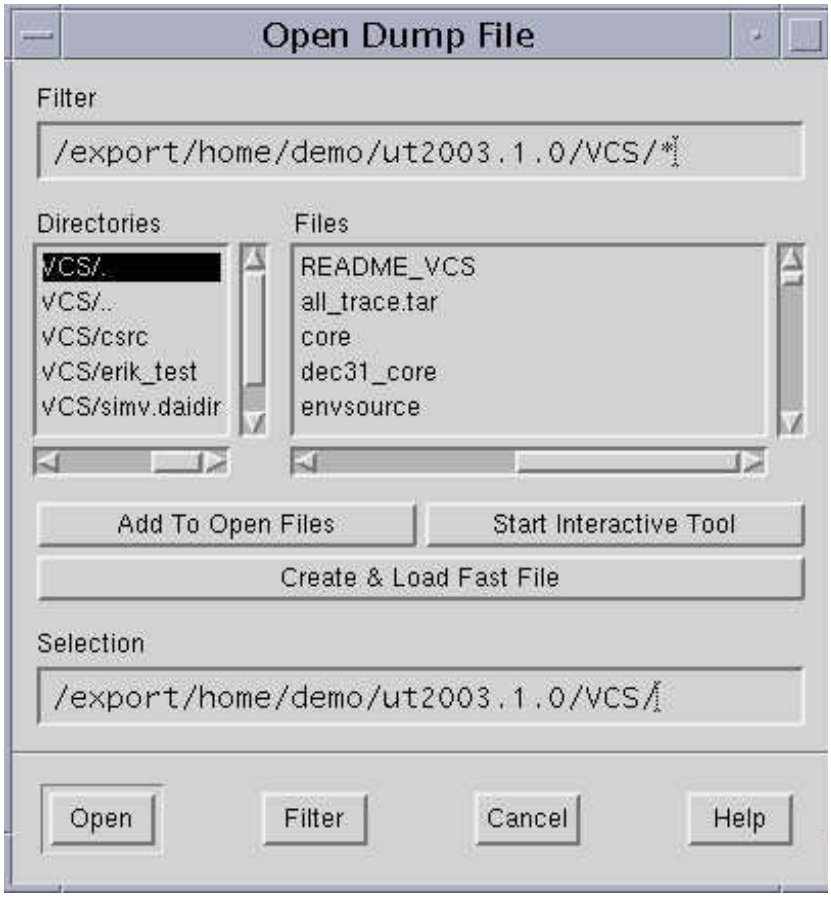


FIGURE 2. Open File Dialog

From this dialog you can use the **Directories** and **Files** fields to select the signal file that you want to load.

- If the signal file is small and can load quickly, you can then simply click **Open** to load only the selected file into Undertow.
- If the signal file is especially large and might be slow to load, then after you select the signal file name, you can click **Create & Load Fast File**. This option converts the selected signal "dump" file to a Fast File format, and then loads this Fast File.

  A Fast File format is a special format that is optimized for Undertow. This file format greatly increases the performance of your Undertow session. Veritools recommends that you use the Fast File format for large files, such as .vce files.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The Fast File format can speed up waveform viewing by over 100,000 times, so you can instantly load and display these Fast Files.

- If you want to load multiple signal files, after you select a signal file name, click **Add to Open Files**. This option pops up a browser window that lets you add additional files to the Undertow session.
- To start the Interactive Suite (the Source Code debugger, and so on) for the selected signal file(s), click **Start Interactive Tool**.

Working With Multiple Files

You can view signals from multiple files. To add signals from one file to another signal file, select the **File => Open => Open Additional File...** menu item.

Undertow lets you view files that use different time scales and even different data formats. For example, your initial file might contain digital signals, and your secondary file might contain analog signals. When you load multiple files, the module browser hierarchy displays a branch for each file opened. Figure 3, "Module Browser - Two Files Loaded" shows two branches, **1** and **2**, representing two different files that have been loaded into Undertow. To view the first file's hierarchy, select **1**; to view the second file's hierarchy, select **2**.

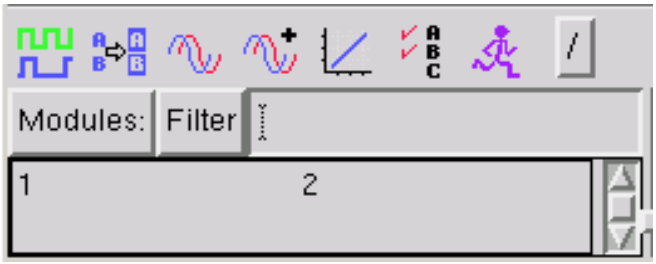


FIGURE 3. Module Browser - Two Files Loaded

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

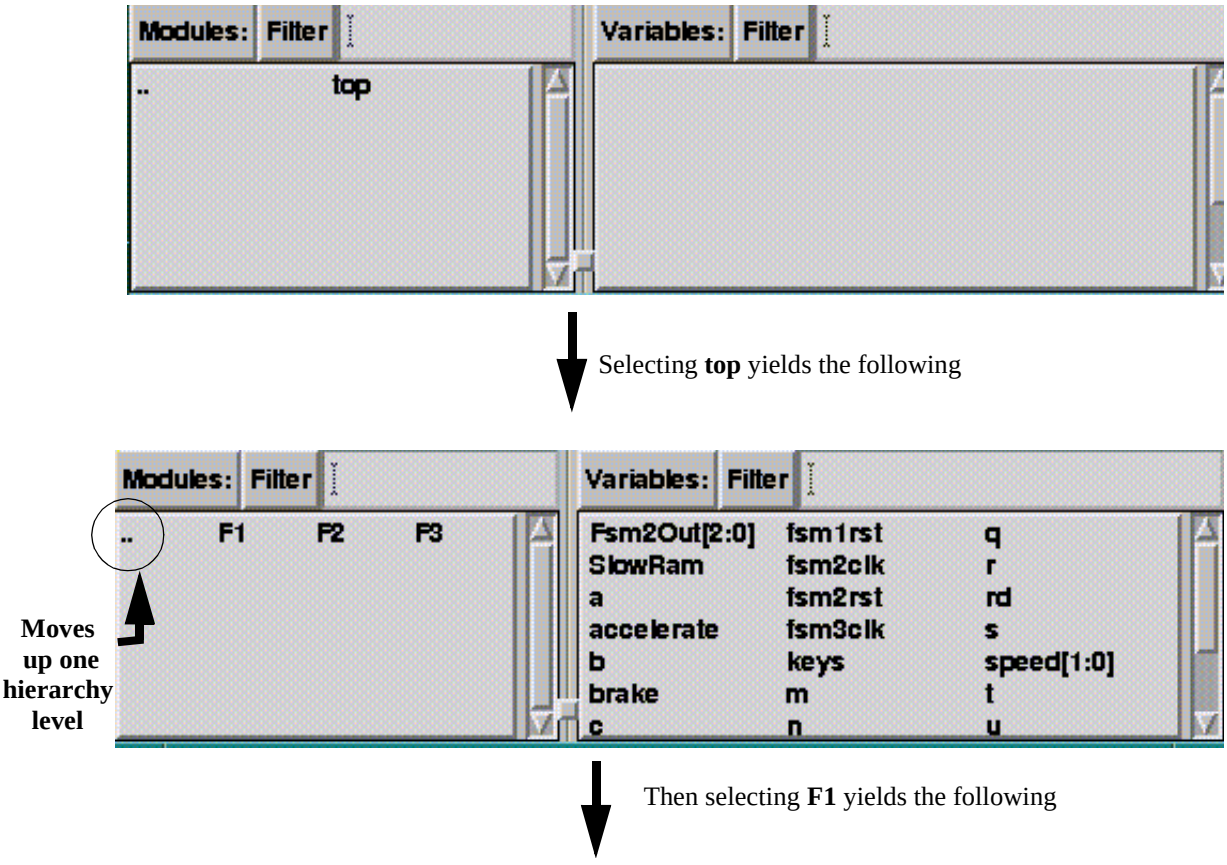
Viewing Waveforms

Chooser

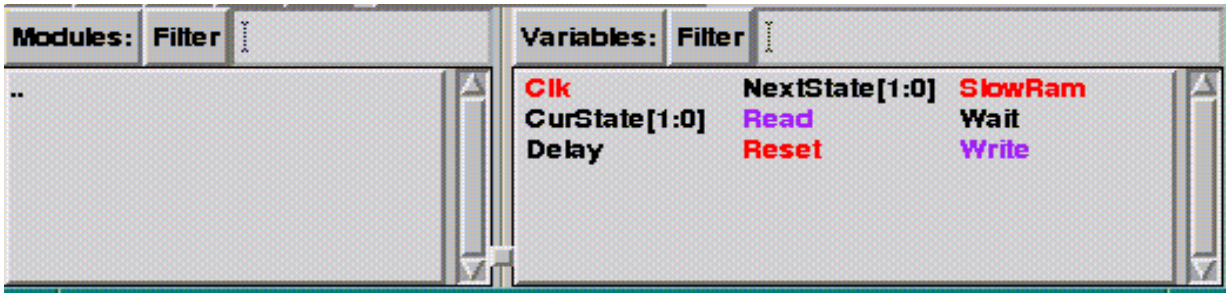
After you load a file into the waveform viewer, you can select from the module and signal browsers (refer to Figure 1, "Undertow Main Window"). These two panes appear when you click the Chooser button, located in the Viewers toolbar.

Module Hierarchy

If you select a module from the module browser, the signal browser displays all signals associated with the selected module. The module browser also displays lower-level modules, contained within the selected module. Figure 4, "Working With the Modules Hierarchy" shows an example of this.



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Red Signals -> Input **Black Signals -> Internal**
Blue Signals -> Output
Purple Signals -> Inputs and Outputs combined

FIGURE 4. Working With the Modules Hierarchy

Viewing Signals

After you select a module, you can use one of three methods to select and view waveforms for a signal or a group of signals:

- Select one or more signals.
- Select all signals.
- Drag and drop signals into the browser.

Figure 5, "Displaying Waveforms" illustrates waveforms for a few selected signals.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

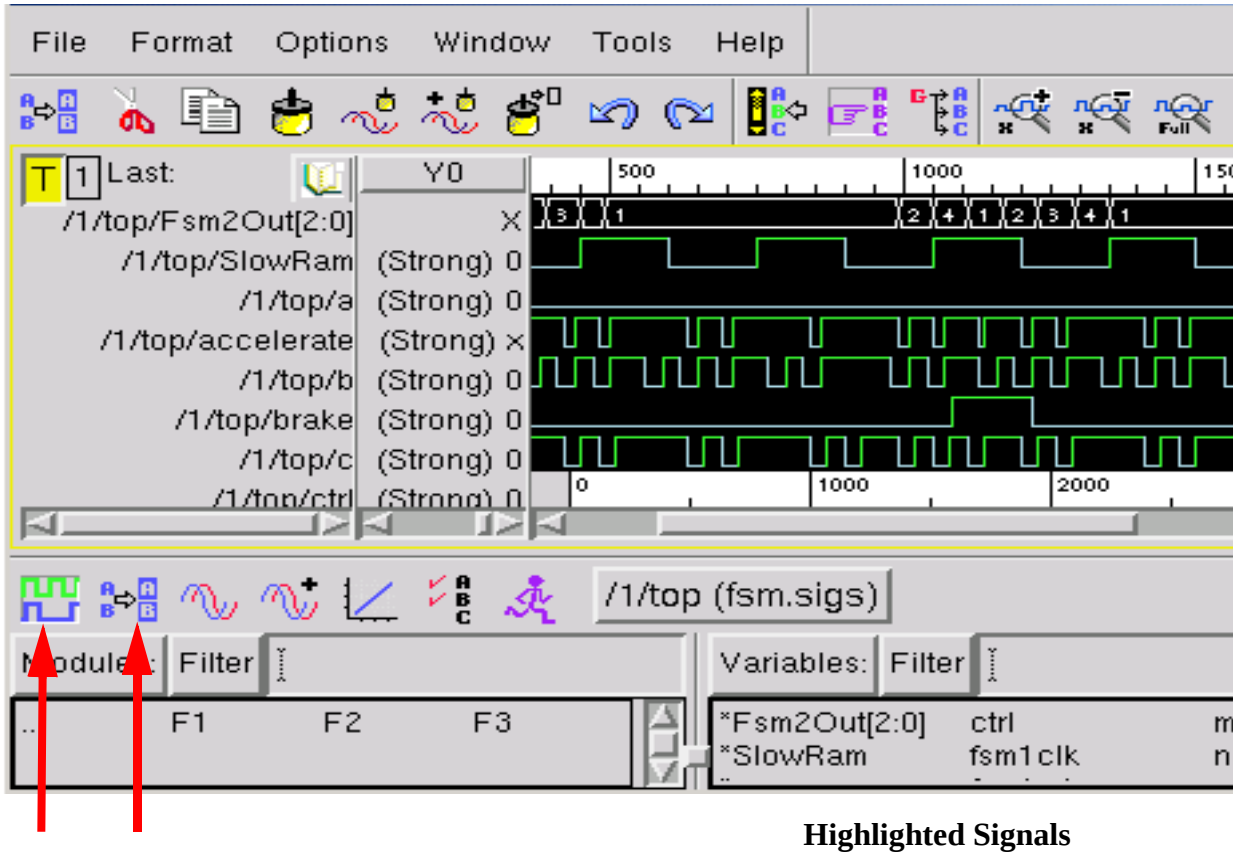


FIGURE 5. Displaying Waveforms

Using The Display Button

1. From the Signal Browser, select the signal(s) to display.
2. Click the **Display** button.

Selecting All Signals To Display

1. Click the **Select All** button.

   Undertow highlights all signals in the signal browser.
2. Click the **Display** button.

Drag And Drop

1. Using the **middle** mouse button, select the **signal** from the signal viewing area.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Hold down the middle mouse button, and **drag** the mouse to the Signal Name area.
3. Release the middle mouse button into the **Signal Name Area** (Figure 1 on page 37).

Zoom Operations

To view more detail of individual waveforms, you can "zoom" into or out of one or more waveforms. The Undertow tool bar includes three zoom buttons, illustrated and described in Table 3, "Zoom Buttons".

TABLE 3. Zoom Buttons

Button	Description
	Zoom In - zooms into the current window, and shows greater resolution of the waveforms displayed
	Zoom Out - zooms out of the current window, resulting in a lower resolution (displays more of the time points on the waveforms).
	Zoom Full - cancels all zooming, and displays waveforms at their lowest resolution.

Using the Mouse to Zoom

Another way to zoom into an area of interest is to use the **right** mouse button.

1. Press the **right** mouse button down on one time boundary of the waveform area to zoom.
2. With the right button down, **drag** the mouse to the other time boundary of the waveform area to zoom.
3. Release the mouse button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The waveform viewer resizes its content to display the area defined by the left and right time boundaries.

You can also establish two time points by **left** clicking **twice**, then **right** clicking.

- Doing this in the waveform field fills the viewer between those two time points.
- Doing this in the lower time scale area zooms to that period of the simulation.

Panning

1. To pan, in the **slide window** (at the bottom of the Undertow window), hold down the **left** mouse button or the **right** mouse button.

   Holding down the left mouse button pans left, and the right mouse button pans right.

2. To control the panning speed, while holding down the left or right mouse button, move the mouse to the left for a slower pan, or to the right for a faster pan.

Setting Waveform Height

To view more detail of a waveform in the vertical direction, you can set the Waveform Height. From the Undertow menus, select **Window => Waveform Height**. The two illustrations in Figure 6, "Waveform Height Set to 20 (top) and 50 (bottom) Pixels" show the same signals, but with different height settings. Undertow specifies waveform height in pixels.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 6. Waveform Height Set to 20 (top) and 50 (bottom) Pixels

Placing Time Cursors

Undertow uses thee time cursors, any of which you can set and make visible at any time (see Figure 7, "Time Cursors").

- To place or move the **T0** cursor, move the cursor to the desired spot in the waveform viewer, and click the **middle** mouse button.
- To place and move the **T1** cursor, hold down the **Control** key while pressing the **middle** mouse button.
- To place and move the **T2** cursor, hold down the **shift** key while pressing the **middle** mouse button.
- To clear all cursors from the display area, select the **Windows => Cursors** submenu from the Undertow menus.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

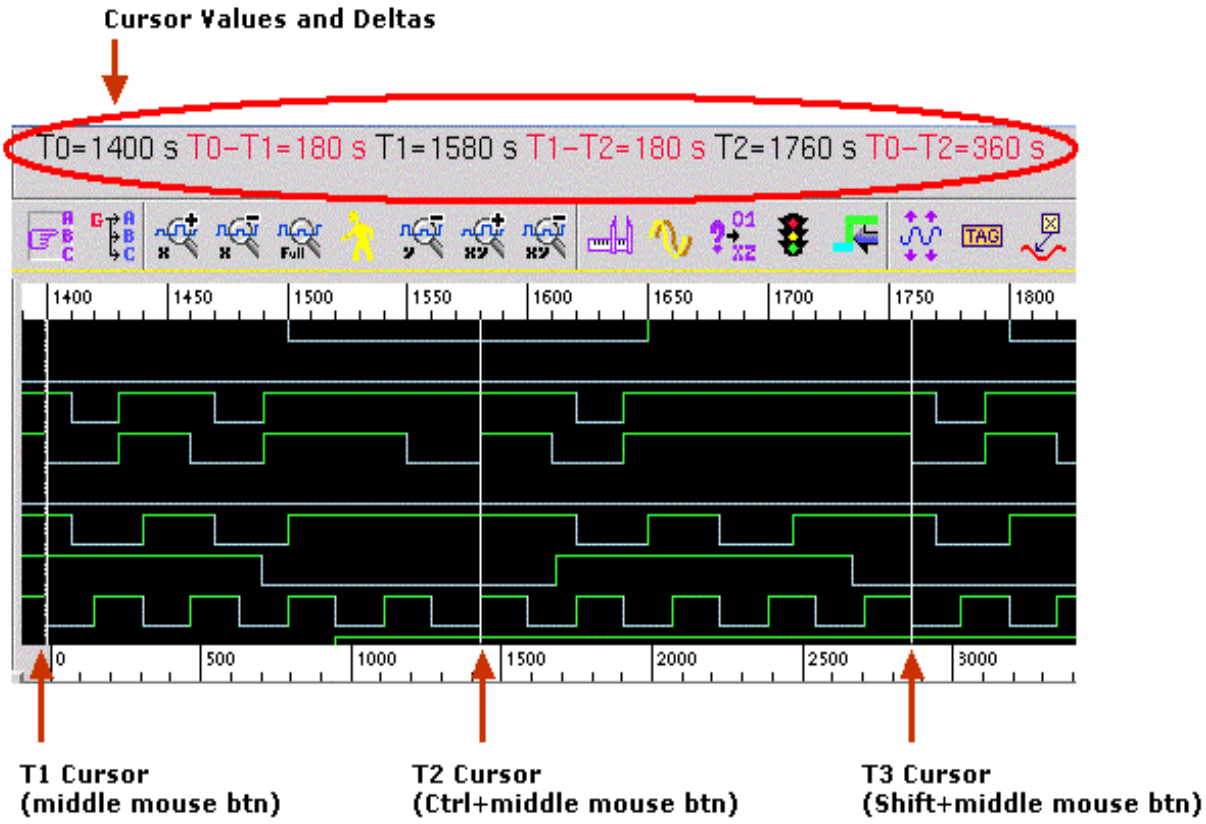


FIGURE 7. Time Cursors

Placing Tags

You can use tags to annotate your waveforms with text, measurements, or values in the waveform viewer. Figure 8, "Tags" displays an example of a **text** tag and a **measure** tag. The third tag type, the **value** tag, is similar to the **text** tag, but uses fixed numeric values instead of text.

Tags are displayed from the **Windows => Tags** pulldown menu, or from the Tag buttons located in Undertow toolbar (3, 4, and 5 buttons from the right).

Undertow pops up a dialog box to indicate how to place these tags. The *measure* tag has two anchor points; the *text* and *value* tags each have one anchor point.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

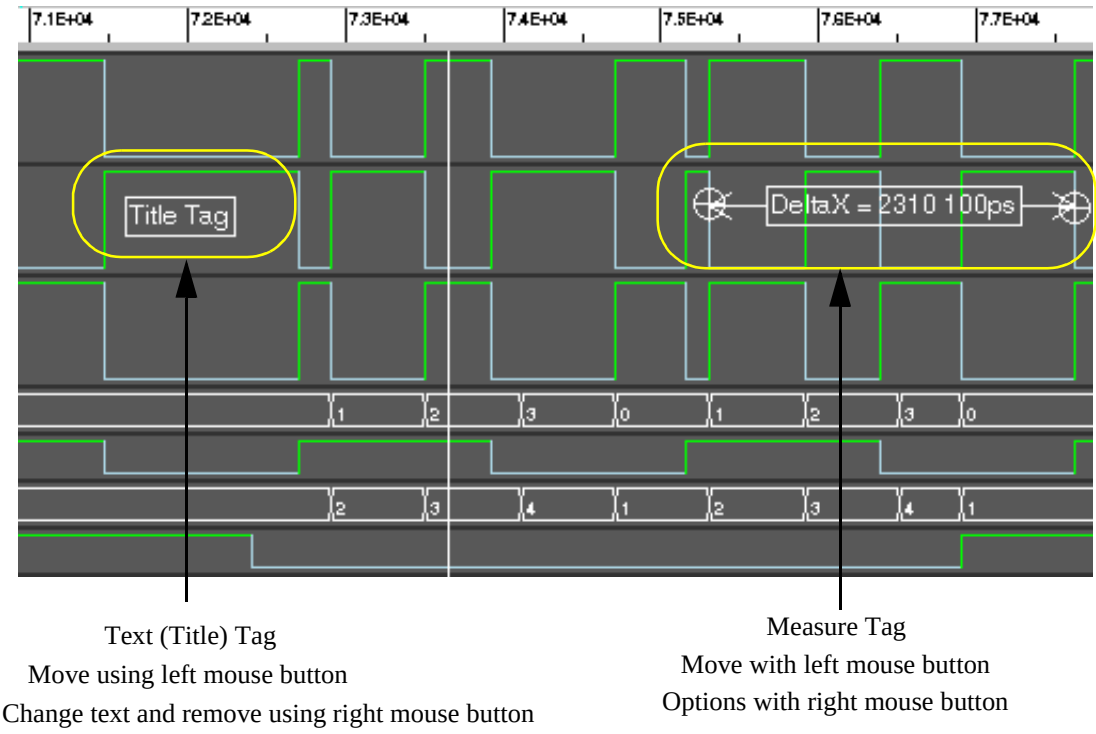


FIGURE 8. Tags

Finding Edges And Values Of A Signal

GoTo Button

To find a value or an edge of a signal, use the **GoTo** feature in Undertow. You can start this feature either from the toolbar button (shown to the left), or by pressing **Control-G** on the keyboard. Either way, the GoTo dialog box opens (see Figure 9, "GoTo Dialog Box").

 Copyright Veritools 2004 All Rights Reserved

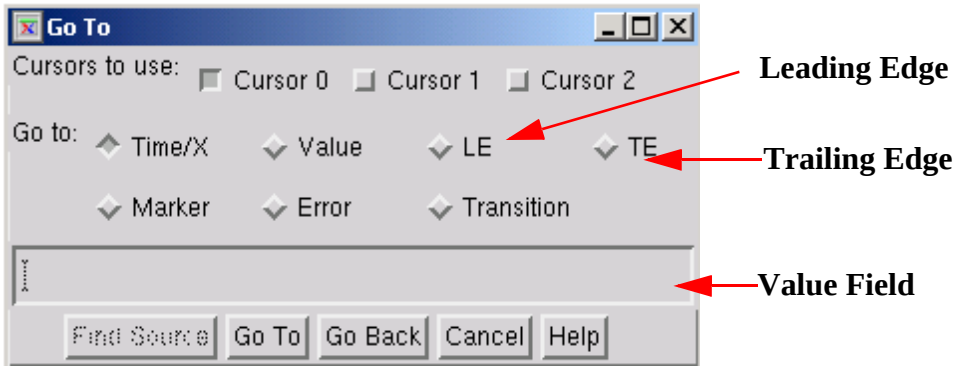


FIGURE 9. GoTo Dialog Box

The GoTo Feature searches for a signal characteristic.

1. In the **Signal Name** area of the Undertow window, enter a signal name.
2. To start the **GoTo** feature, either use the toolbar button, or press **Control-G** on the keyboard.
3. In the GoTo dialog, select the cursor to use (**T0**, **T1**, or **T2**).

   See Section , "Placing Time Cursors".
4. Select the type of search to perform.
   - To search for a value, fill in the **Value Field.**
   - To perform a forward search, select the **Go To** button.
   - To perform a backwards search select the **Go Back** button.

Adding Panes

Undertow can display multiple waveforms in one pane, or in multiple panes.You can add as many panes as you need in Undertow. For example, you might want to display the waveforms for different modules in separate panes, as illustrated in Figure 10, "Multiple Panes".

To add or remove panes in Undertow, use the **Windows** menu. This menu list two options for adding panes. Table 4, "Adding Panes" describes these two pane options.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 4. Adding Panes

Pane Type	Description
Time	Display pane with X axis as time
Arbitrary X	Display pane with X axis defined by file information.

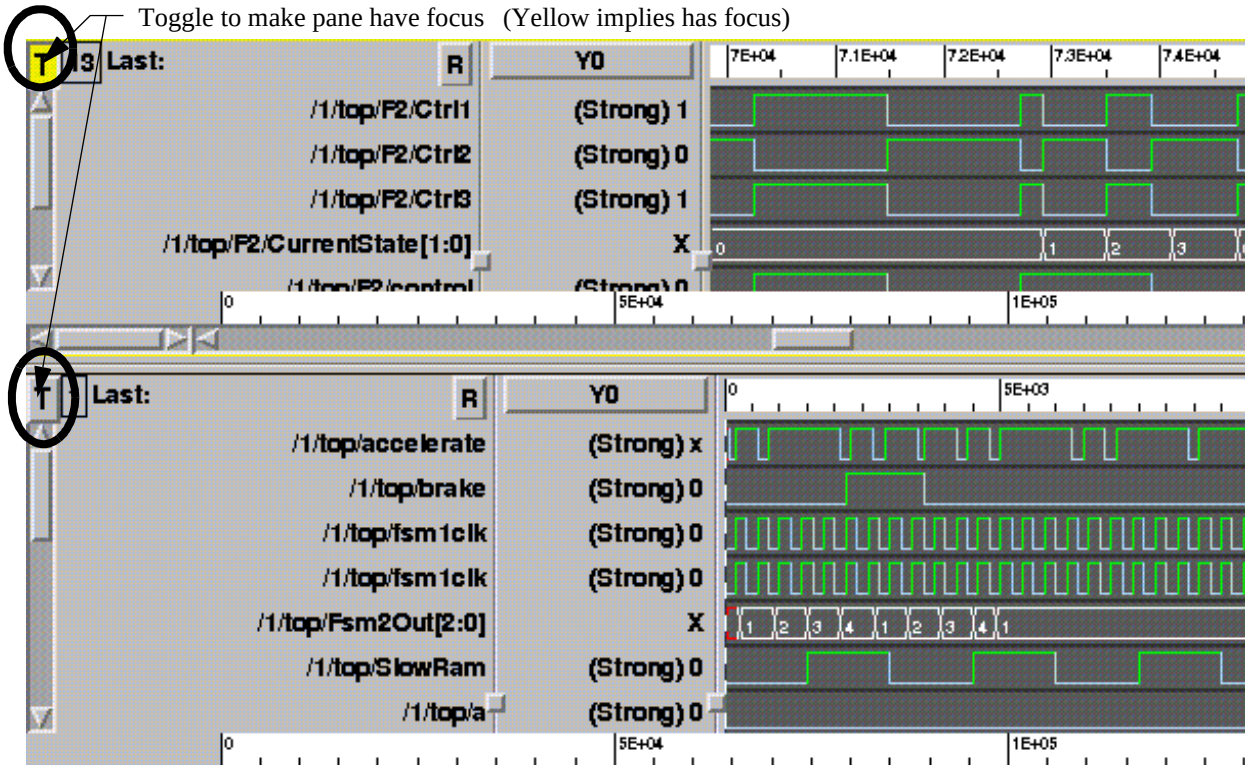


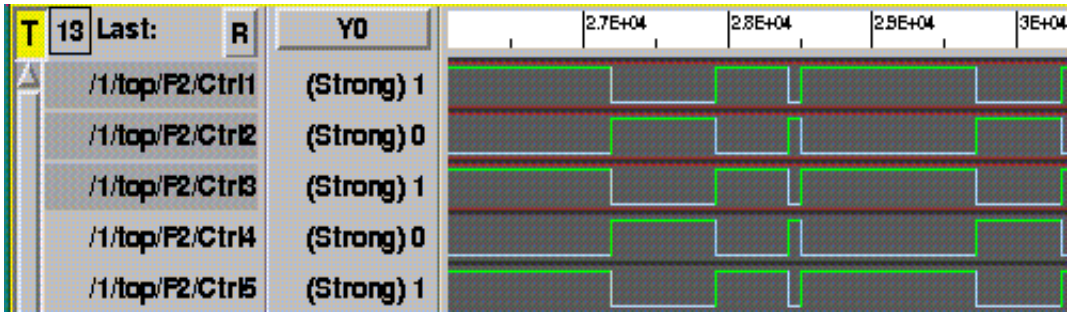
FIGURE 10. Multiple Panes

Bundling Signals

You can *bundle*, or group, two or more signals into a new signal. Undertow displays the grouped signal as a bus, which is the sum of the selected signals. Figure 11, "View Of Signals Before Bundling" shows three signals, **Ctrl1, Ctrl2**, and **Ctrl3**.

FIGURE 11. View Of Signals Before Bundling

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



1. To bundle these signals, select **Options => Bundle => New** from the Undertow menus.

   The Bundle Signals dialog box opens, as shown in Figure 12, "New Bundle Dialog".

2. Drag the selected signals into the **Signals to Include in Bundle** area in the dialog box.

   In this example, the selected signals are **Ctrl1**, **Ctrl2**, and **Ctrl3**.

3. In the **Bundle Name** field, name this new bus.

   In this example, the bus is named **NewBundle**.

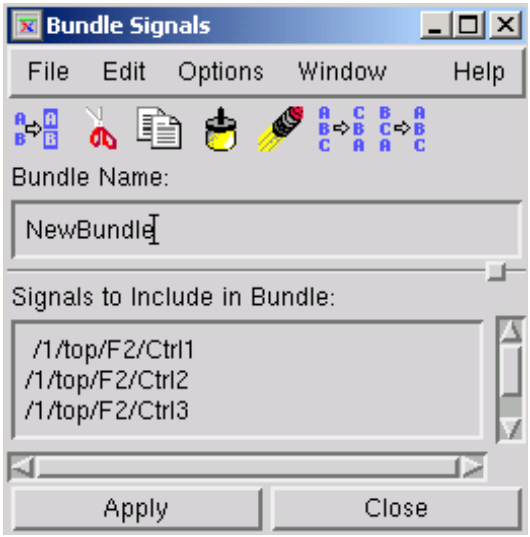


FIGURE 12. New Bundle Dialog

4. Press the **Apply** button in the dialog box.

   Undertow displays the bundled waveform (see Figure 13, "Bundled Signals").

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

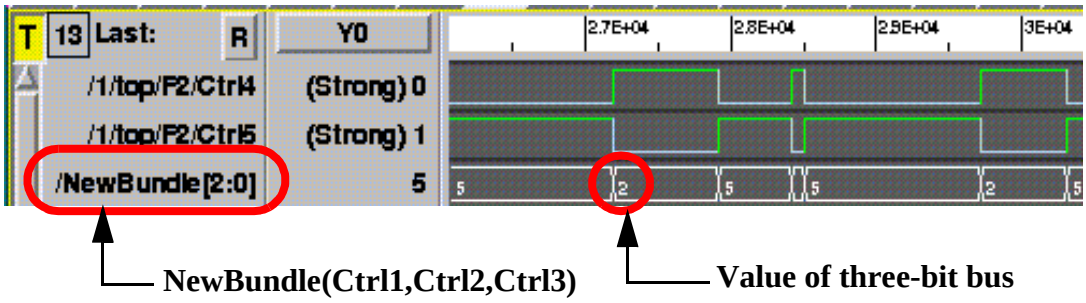


FIGURE 13. Bundled Signals

Working With Aliases

You can use aliases to define names and values for buses. For example, an alias can help you when you analyze a bus for a specific value. Figure 14, "Define Alias For Value" contains a two-bit bus named *speed*. When both bits are positive, the value for the *speed* bus is 3 (2**1 + 1 = 3).

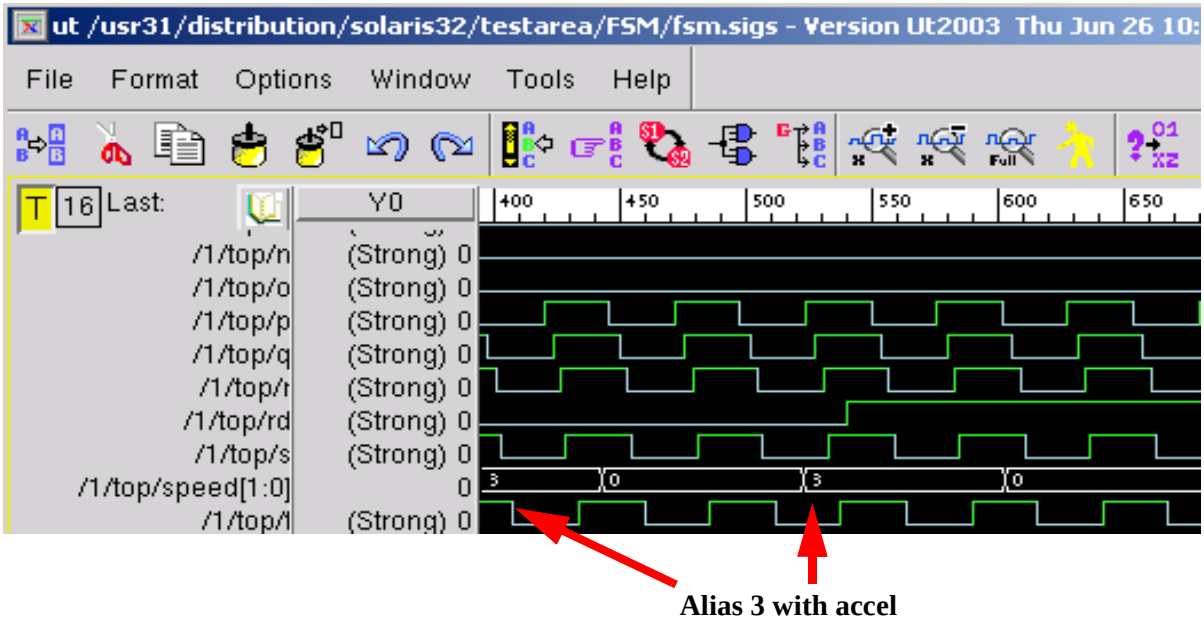


FIGURE 14. Define Alias For Value

If a value of 3 for the *speed* bus implies acceleration, then you can define a valued alias for this bus value.

 Copyright Veritools 2004 All Rights Reserved

1. (Optional). Select one or more signals to assign to the alias.
2. From the Undertow menu, select **Options => Alias => Value => New**.

   This menu choice pops up the dialog box shown in Figure 15, "Alias => Value => New Dialog".

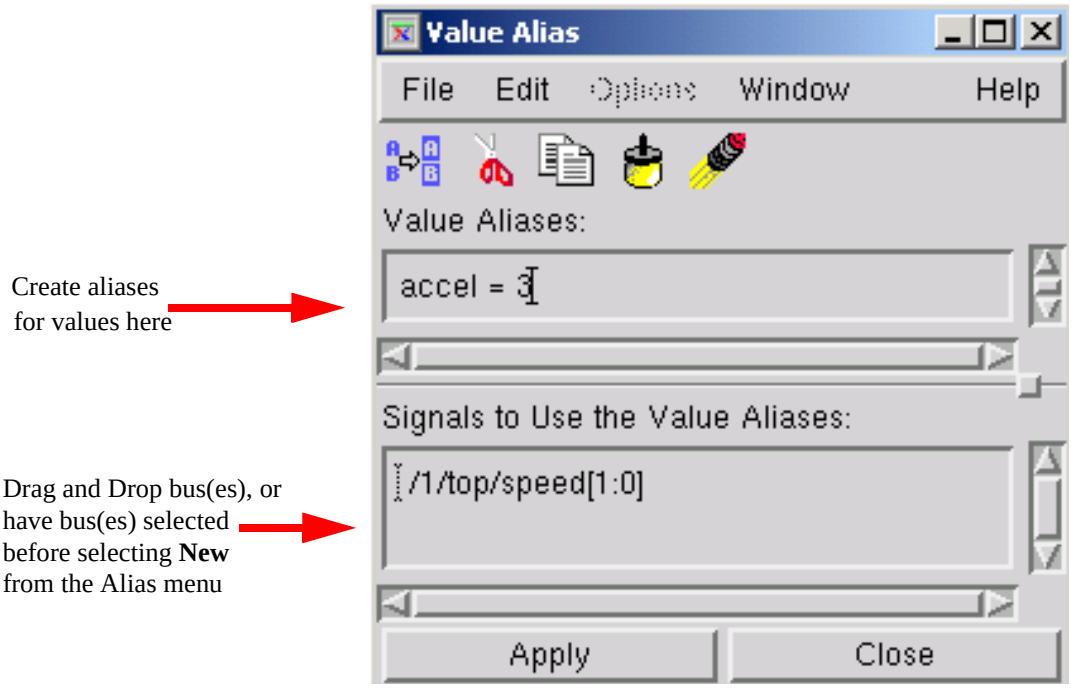


FIGURE 15. Alias => Value => New Dialog

3. In the **Value Aliases** field, assign a name and a value to the alias.

   For example, accel = 3 indicates the alias name is accel and the value is 3.
4. If you did not select signals in Step 1, drag and drop one or more buses into the **Signals to Use the Value Aliases** field.

   For example, speed[1:0] assigns the accel alias to a two-bit bus named *speed*.
5. Click **Apply**.

   Figure 16, "Aliases Displayed" shows the changes to the waveform viewer when you apply an alias.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

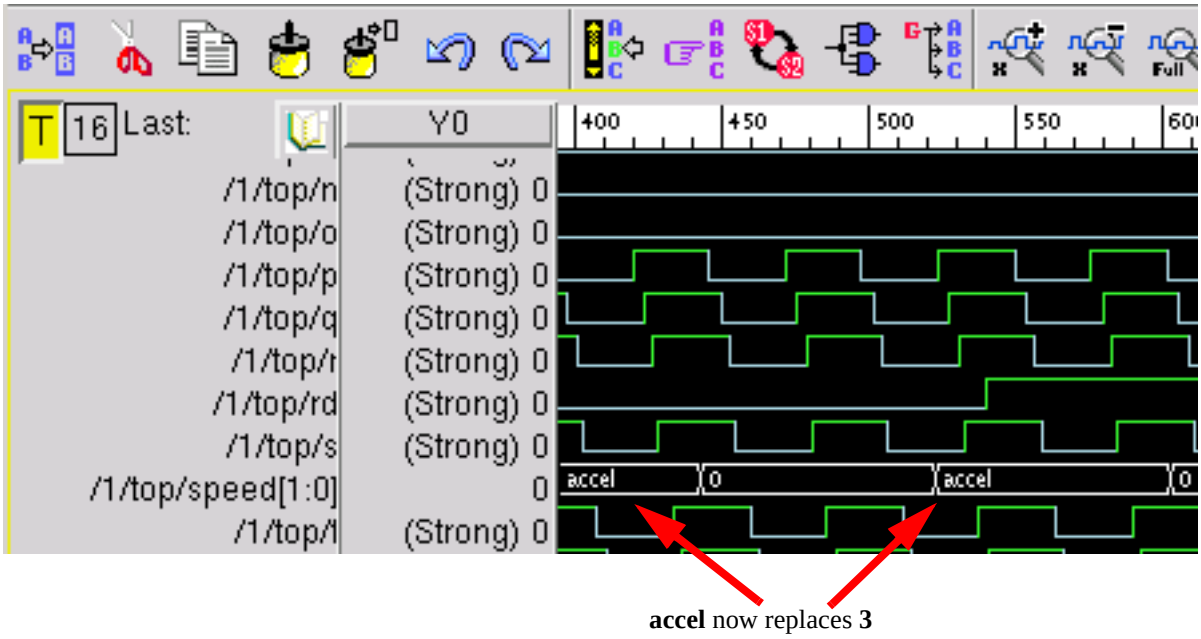


FIGURE 16. Aliases Displayed

6. After you assign the alias, you can use the Undertow **GoTo Tool** to search the *speed* signal, which uses the accel (acceleration) alias.

   Figure 17, "Using an Alias in a GoTo" shows the **accel** alias for the *speed* bus, used in a GoTo dialog box. For more information about this dialog box, see Section , "Finding Edges And Values Of A Signal".

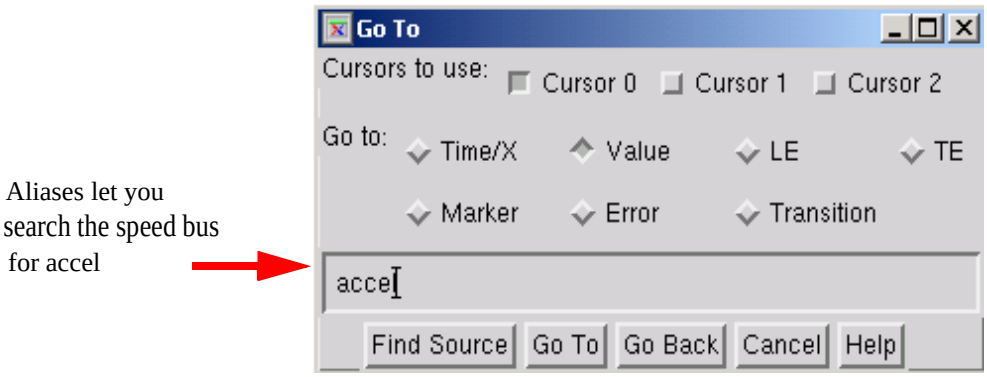


FIGURE 17. Using an Alias in a GoTo

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Description

Undertow is a powerful waveform display tool for viewing digital, mixed mode, and analog simulation output from a variety of simulators, including:

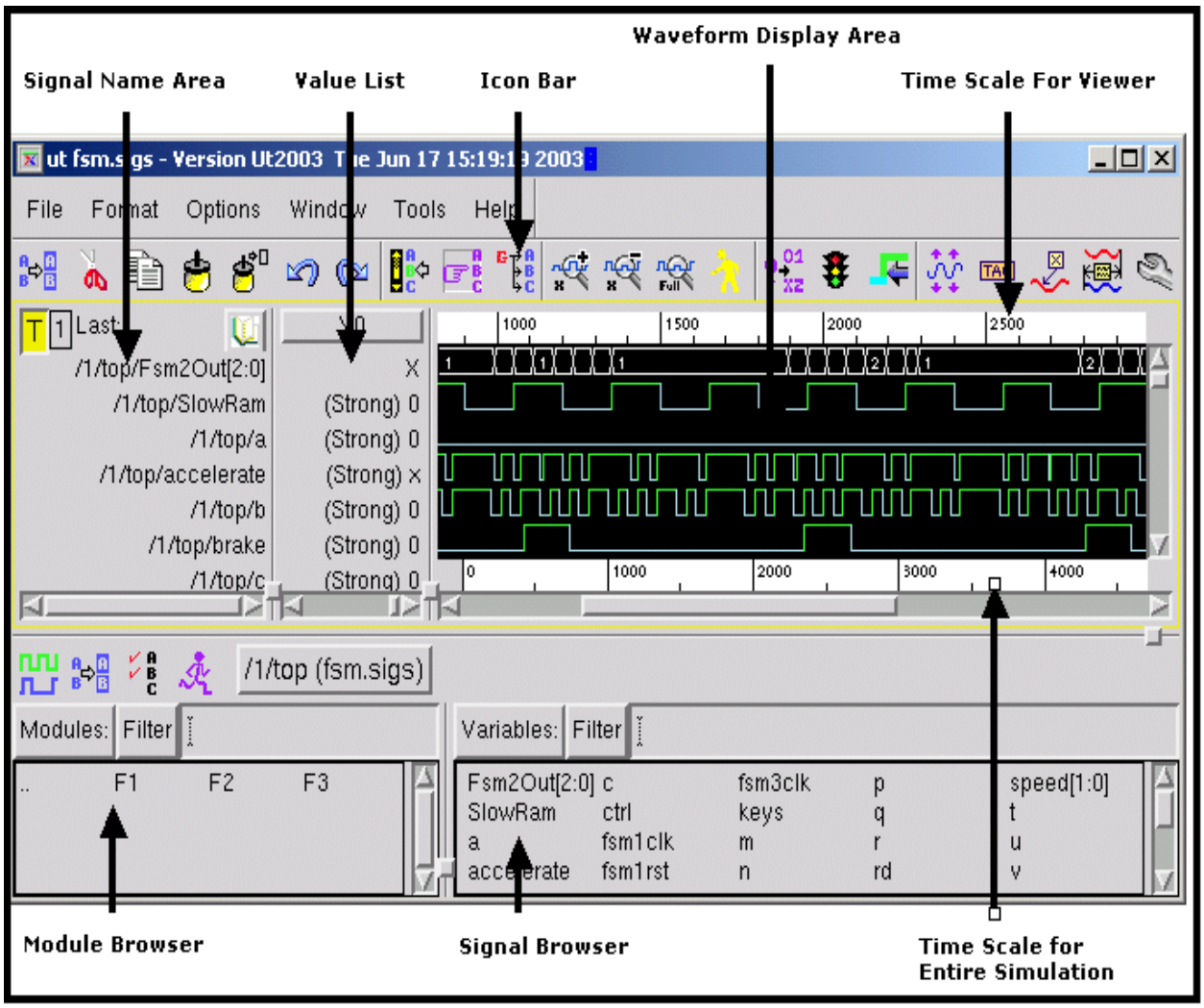
- Verilog
- TIMEMILL
- POWERMILL
- HSpice
- VHDL (VSS, LEAPFROG, MENTOR, and ModelTech)
- Anagram

You can use Undertow to:

- View output files interactively, in batch or real-time.
- Trace netlist connectivity.
- Compare files.
- Toggle analysis capabilities.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 35. Undertow Main Window



Menu Bar

The Undertow menu bar is located at the top of the **Display** window. To tear off (pin down) an individual pull-down menu from the Main Menu, click the **menu dashed line.**

The complete description of each feature listed in the menu can be found below. You can also use the ***Windows => Floating windows*** menu option to detach the Main Menu from the main Undertow window (see Section , “Main Toolbar Icons”).

 Copyright Veritools 2004 All Rights Reserved

Cursor Time Values

Undertow provides three Time Cursors. The value of the Time Cursors, and the delta between them, is listed at the top right of the window (see Section , “Time Cursors”).

Time Markers

You can place an unlimited number of time markers anywhere on the Wave Display. To set a time marker:

1. Place the cursor at the desired spot on the Wave Display (use the Snap On to place it on an edge).
2. Hold the **Shift** key and click the **left** mouse button.

   The GoTo window pops up, and shows the time value for the marker.
3. *(Optional).* You can add comments to the time value.
4. Click **Add** to include the new marker in the list. (see the description of the **GoTo** icon in Section , “Main Toolbar Icons”, or from the main window Help).

Icon Bar

You can select commonly-used features without clicking in a menu.

- The buttons (**Fast Selection Menu**) for these features are below the **Main Menu Bar**.
- To see a text description for each icon, move the cursor over the desired icon. The description displays on the right side of the button row.
- Each button is associated with a hot key, which you can modify in the application defaults file.
- You can use the ***Windows => Floating windows*** menu option to detach the **Fast Selection Menu** from the main Undertow window.

Signal Name List

The Signal Name List is the left column of the waveform display (main Display) window. This column shows all or part of the selected signal names.

1. To view the whole signal name, hold down the **right** button on any signal.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

A small pop-up window shows the whole signal name. This window disappears automatically.

2. If you need long signal names, use the square button (at the lower right corner of the window) to resize this column.

3. To change the initial font size or width of the column, make the appropriate changes to the Undertow file, or click ***Options*** => ***Properties*** => ***Fonts*** to change the fonts on the fly.

To move signals around the display, do either of the following:

- Use the **middle** mouse button to **select and grab** a signal, then drop that signal anywhere within the current viewing area of the Signal Name List to move the signal to a new location, or

- Highlight the signal, then use the **Cut** or **Copy** icon to move the signal. Scroll to where you want to move the signal. You can left click a signal in the list. to place the moved signal on top of the selected signal. Or you can use the **Paste** icon to add the moved signal to the end of the list.

To record the values in the **Value Area/Areas**, click **R**.

*Note: You can create as many value areas as you want. For each value area, click the **right** mouse button at the top of that area, to set that area to a different value.*

Value List

The Value List is the middle column of the main Display window. This column shows a value for each signal at the Time Cursor 0 (T0) position.

- To change the format of the value, use the Format menu.

- To view the format of a value, hold the **right** button down at **any value**.

  A small pop-up window shows the whole signal value and its format. This window disappears when you release the **right** button.

- If you need long values, use the square button (at the lower right corner of the window) to resize this column.

- To change the initial font size or width of the column, make the appropriate changes to the Undertow file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Wave Display

The Wave Display is the main portion of the Undertow Display window. To modify how many waves initially display on any screen, or the pixel height of any signal, edit the Undertow file.

Time Cursors

You can place time cursors anywhere on the Wave Display. Undertow provides three cursors.

- To set the first time cursor, place the cursor at the desired spot on the Wave Display, and click the **middle** mouse button. You can use the ***Window => Fix Cursor*** menu option to attach this first cursor to either the signals or the Display Window itself.
- To set the second time cursor, hold the **Control** key while clicking the **middle** mouse button.
- To set the third time cursor, hold the **Shift** key while clicking the **middle** mouse button.
- To place any time cursor exactly on the nearest edge of the selected signal, toggle the **Snap** button in the **Fast Selection** menu.

Time Scales

The Display window provides two Time Scales.

- The lower Time Scale shows the whole time of your simulation.
- The upper Time scale shows the visible time of your simulation.

You can set Zoom markers on the lower time scale, so you can Zoom In to an area outside of the current display area. To set the Zoom markers, **left** click twice to **define the zoom area**, then right click to end the area selection.

Scroll Bars

The Undertow window provides two Scroll Bars:

- The horizontal scroll bar is located at the bottom of the Wave Display.
- The vertical scroll bar is located at the left of the Signal List.

To move the scroll bars, use the **middle** mouse button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To pan the windows, place the cursor on the arrows at the extremities of the scroll bar areas.

- Click the **left** mouse button to pan the display in the direction indicated by the arrow.
- Click the **right** mouse button to pan the display in the opposite direction.

The horizontal scroll bar includes a **smooth scrolling mode**. To activate this mode, hold the **left** or **right** button while the cursor is on the scroll bar area. The speed of the smooth scroll is determined by the horizontal position of the mouse arrow.

- Move the mouse arrow to the left to slow down the scrolling.
- Move the mouse arrow to the right to accelerate scrolling.

When you drag the vertical scroll bar, the window next to the active pane button (top left) indicates the position of the signal at the top of the display window.

Main Toolbar Icons

TABLE 5. Main Toolbar Icons

Icon	Description
	Select All (Ctrl-A)—selects all of the signals that are currently displayed.
	Cut (Ctrl-T)—clears the display from the selected signals, and saves them in the clipboard. Pressing the Cut key on the (Sun) keyboard has the same effect.
	Copy (Ctrl-Y)—copies the selected signals to the clipboard. The selected signals remain on the display, but are “unselected”. Pressing the Copy key on the (Sun) keyboard has the same effect

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description
	Paste (Ctrl-P)—pastes the contents of the buffer, either at the bottom of the display or above the previously-selected signal. Pressing the Paste key on the (Sun) keyboard has the same effect.
	Paste Blank—inserts an empty row into the Waveform viewer, to separate signals
	Undo—Undo the last action performed on a the Waveform viewer
	Redo—Redo the last action performed on a the Waveform viewer
	Scroll To (Ctrl-V)—opens a dialog window, that you can use to build a list of signals. The vertical scroll routine uses the signals in this list. (you can also select ***Vertical Scroll*** **=>** ***Scroll To Signal*** from the window menu).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description
	Choose (Ctrl-C)—appends or removes the Choose window at the button of the Display window. • To adjust the size of the Choose window, use the button located under the horizontal scroll bar on the right side. • To open the Choose window as a floating window, select the ***Window*** **=>** ***Floating Choose Window*** menu option. This window has three main areas: • menus • module list scroll window • signal list scroll window You can select signals in one of three ways: • To select a signal from the signal list, place the cursor on the name and click the **left** mouse button. • To select a group of signals listed consecutively, select the first name, then hold the **Shift** key while you select the last name in the sequence. • To select signals names from different areas of the list, select the first name, then hold the **Ctrl** key while you select each additional signal name. Click the **Display** icon to display the value of the variable and its format at the time T0 (first time cursor).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description
	Groups (Ctrl-M)—appends or removes the group window at the right side of the waveform display window. You can use this window to define, modify, or display groups of signals. • This window contains a scrollable window and a button to add new group definitions to the list. To detach the groups window from the Display window, use the Floating Groups Window feature (from the Main Menu Bar, select ***Window => Floating windows***). Please refer to 5.3.1 for more details on Grouping
	Zoom In (Ctrl-Z)—increases the zooming factor by one step, or fits the window to the given zoom markers. To see the zoom markers: 1. Place the mouse arrow at the desired boundaries of the window, or at the bottom time scale. 2. Click the left mouse button. The chosen marks are shown with x. Another way to achieve the same effect is to click the **right** mouse button when the cursor is inside either the waveform display area or the bottom time scale.
	Zoom Out—Performs a zoom out of 200% on the Waveform viewing area.
	Zoom Full (Ctrl-F)—shows the entire simulation time window.
	Waveform Actions—pulls down a menu that contains many functions for waveforms.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description
	Update—Performs a a screen refresh. Use this icon to display a waveform in real-time mode, so you can load in your latest data.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description

	GoTo—Pops up a dialog box that you can use to navigate to a specified point of interest. For all options:
	• The Go To button finds the next occurrence, after time cursor T0.
	• The Go Back button finds the previous occurrence, before time cursor T0.
	The options are:
	• Time—moves the first time cursor (T0) to the time value that you entered in the input field.
	• Value—moves the first time cursor (T0) to the signal value that you entered in the input field for the signal selected on the Display window.
	The value can be an alias. If the signal is a bus, the value needs to be in the same format as displayed in the signal values area.
	• Error—moves the first time cursor (T0) to the area where errors have been highlighted (plum colored background) on the selected signal. You can use this feature after you run the Test Analyzer, or Find Logic from the Logic Analyzer.
	• LE—moves the first time cursor (T0) to the next leading edge of the selected signal.
	• TE—moves the first time cursor (T0) to the next trailing edge of the selected signal.
	• Transition—moves the first time cursor (T0) to the next transition of the selected signal/bus.
	• Marker—moves the first time cursor (T0) to the next marker, or to the marker highlighted in the list below the input field. If you press the Shift key while you click the left mouse button, the Goto window opens with the Marker option activated.
	  • Use the input field to view or edit the value where the marker was placed.
	  • Use the read-only window, directly below the input field, to view all defined markers, and to select the marker to use with the **GoTo** feature.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 5. Main Toolbar Icons (Continued)

Icon	Description
	Snap—Snaps the selected cursor to the nearest edge of a selected signal. **Not Shown: Snap Source**. Synchronizes the Source Code window to the cursor.
	Fit To Pane—Adjusts (expands) the size of the selected signal(s), to fill the height of the Waveform viewing area. To cancel this operation, click the middle mouse button in the signal area.
	Tag—Lets you annotate Waveform viewer (see Section , “Tags” for more information).
	Value Tag—Value tags contain the time value for the chosen point. (see Section , “Tags”).
	Measure Tag—Similar to value tags, but these tags contain a delta time value, reflecting the distance in time between two points.
	Analog Configuration—opens the analog properties window (see Section , “Application Note - Analog Features”).

Table 5, “Main Toolbar Icons” shows most of the Toolbar option buttons, including analog features.

- To toggle the analog functions on or off, enable or disable ***Options*** **=>** ***Analog Features***.
- Two additional features not described in Table 5, “Main Toolbar Icons” are the **State Tool** and **Schematic Tool** buttons. These icons open the **State Diagram Tool** and the **Schematic Window Tool**, respectively.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Grouping Signals

FIGURE 36. Group Windows

General Descriptions

Use this window to define, modify and display groups of Signals

Groups Window

Groups (**Ctrl-M**) appends or removes the group window at the right side of the wave display window. Use this window to define, modify and display groups of signals. To detach the Groups window from the Display window, use the **Floating Groups Window** feature (see ***Main Menu Bar => Window => Floating windows)***. It contains a scrollable window and a button to add new group definitions to the list. The groups window has the following buttons:

- **Clear Last Group**—clears the last group displayed. This does not clear the group's definition. You can display the same group again later.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Clear All Groups**—clears all groups displayed. This does not clear the group's definitions.
- **Delete**—switches to delete mode. If it is on (pushed), clicking on a group button deletes its definition.
- **Display**—switches to display mode. If it is on (pushed), clicking on a group button adds it to the waveform display area.
- **Modify**—switches to modify mode. If it is on (pushed), clicking a group button pops up the Modify Group window. This window has an input field for the Group Name, which you can use to modify the group name. The **Signals to Include in Group** scrollable window lets you edit the names of the signals included in the selected group. The **Modify Group** window has the following buttons:
  - **Paste**—appends the signals from the buffer to the Modify Group window. You can cut/copy signals from the Waveform Display window or from the Choose window.
  - **Blank**—inserts a blank line at the cursor location (it actually adds BLANK to the **Signal to Include in Group** window, which is then displayed as a blank line on the waveform display window).
  - **OK**—applies the modifications and dismisses the **Modify Group** window.
  - **Clear**—clears the **Signal to Include in Group** window.
  - **Cancel**—cancels the modifications and dismisses the **Signal to Include in Group** window.
- **Help**—opens the related help window.
- **Group Definition**—each group definition is represented by a button in the scrollable list. The first 26 group buttons have a label that starts with a capital letter and a colon. To access these group definitions, use a bind key combination: hold the Shift key and click the capital letter corresponding to the desired group.
- **New Group**—Use this button to create new groups definitions. When activated, the **Modify Group** window comes up and lets the user enter the new group's definition.(For more details, please refer to the section of Creating/Modifying Groups)
- **File Group**—Use this button to save or display a list of signals as a group. Clicking this button opens the **File Selection** dialog box.
  - If you click the **Modify** button, then you must enter a file name and click **Save**, to save the list of highlighted signals into the file.
  - If you click the **Display** button, then you must select the file and click **Load**, to display the list of signals as a group.

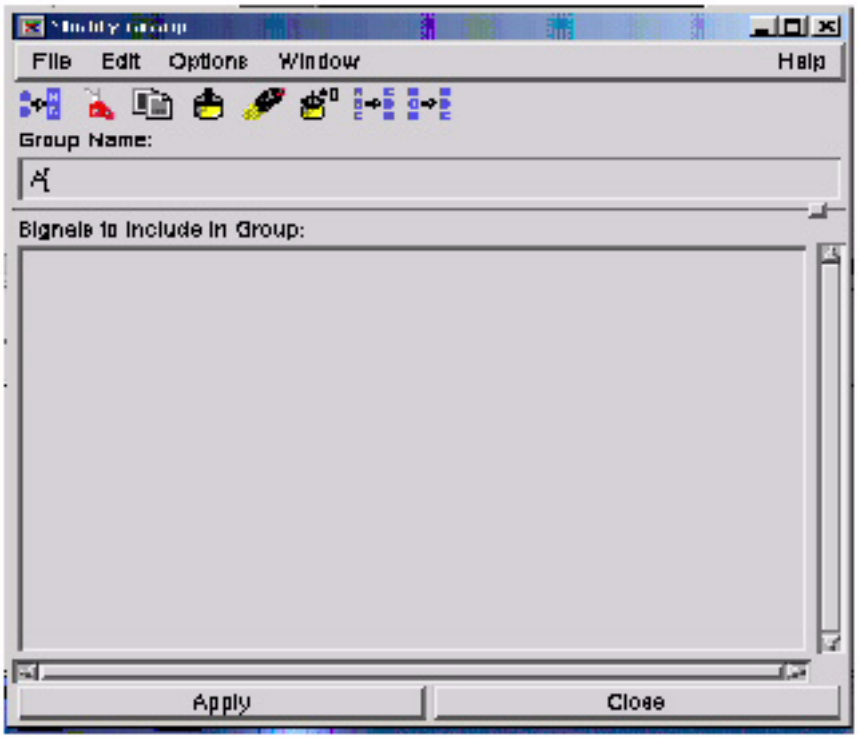
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

*Note: The file that this feature creates is a list of signals, not a group file as created with **File** => **Save** => **Save Groups**. If you use **File** => **Restore** to load the file, Undertow displays the same signals, but not as a group.*

- **Saving groups to a File**: After you define the groups to save, select ***File*** => ***Save*** from the menus. The **Save** dialog box opens; use this dialog to specify a filename for the groups file. To save the groups, click the **Save Groups** button.

- **Restoring groups from a file**: Select ***File*** => ***Restore*** from the menus. Select the file containing the groups information (generated using the **Save Groups** feature), and click on **Restore**. Undertow adds the saved group definitions to the list of existing group definitions.

Creating/Modifying Groups



Menu

File: Open - ope a alias File

SaveAs/Save - save the data currently in the alias window.

Edit - Select all/ Cut /Paste/ Clear Texts

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Window - Configure** - To configure the **Text Choose** Window, select the "**Configure** option from the Window pulldown menu.

Icon Menu

- **SelectAll/Cut/Copy/Paste** - select/Cut/Copy/Past text from one text window to another text window
- **Blank** - add a blank line between the group signal. If you add a blank line in between 2 signals, the waveform viewer will display a blank line between that 2 signals
- **Reverse** - sort the signal in the "Signal to Include in Group" window in the reverse alphabetical order.

Sort - sort the signal in the "Signal to Include in Group" window in alphabetical order.

Chooser Tool Bar Icons

If you toggle on the **Chooser** icon (located in the main tool bar), the **Chooser** area displays at the bottom of the Undertow **Waveform** window.

*Note: When you use the Undertow Suite, this display appears in the **Source Code** window.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

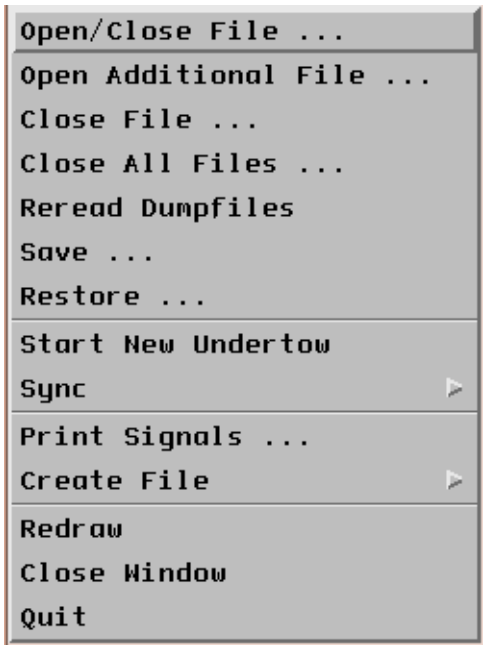
TABLE 6. Chooser Tool Bar Icons

Icon	Description
	Display—To load signals into the Waveform window: 1. Select one or more signals in the **Signal Browser** window. 2. Toggle on the **Display** icon. If no signals are selected when you toggle on the **Display** icon, then the next signal that you select, immediately displays in the **Waveform** viewer. *Note: If more than one panel is open, the selected signals display in the active panel.*
	All—Selects all signals in the **Signal Browser** window. For example, if you click **All** and then click **Display**, Undertow loads all signals into the **Waveform** viewer. To cancel the "all" feature, toggle off this icon.
	Options—See Section , "Options Menu".
	Actions—Shows a popup menu for actions that you can perform in the current context. For example: **Copy** —copies the current selection to the buffer (for use with Paste). The (Sun) Copy key has no effect in this window. **Text**—lets you manipulate groups of signals within a text window. **Explode Bus**—Displays each bit in the selected bus, as a separate signal.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Menu

This section describes the Undertow File pulldown menu.



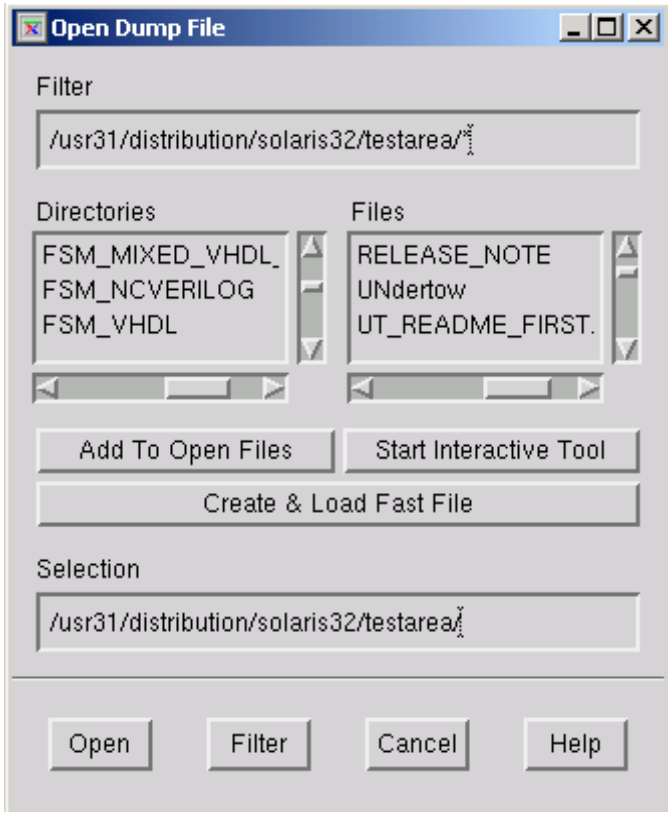
Open Additional File

To open *dump* (signal) files from the Undertow pulldown menus, select either ***File*** => ***Open/ Close File*** or ***File*** => ***Open Additional File***. Figure 37, “File Dialog Box”shows the file dialog box that opens for these menu items.

- **Open/Close File** prompts you whether to unload the files you have currently loaded in your viewer, before loading the new files.
- **Open Additional File** adds files to the file(s) that you currently have open.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 37. **File Dialog Box**



Creating and Loading a Fast File

To open a generic output file (VCD, Timemill, Powermill or Hspice-WSF) and create a *fast file*, invoke the **vdump** program. If the file is compressed, Undertow uncompresses the file and then creates the *fast file*.

The Undertow Suite uses the file that you specify. If you specify a VCD file, the Undertow Suite converts this file to a fast file. If you previously overwrote the VCD file, use one of the following menu options to reload the new file:

- ***Source Code Window*** => ***File*** => ***Open*** => ***Waveform*** or
- ***Undertow window*** => ***File*** => ***Open*** or
- ***Source Code window*** => ***Actions*** => ***Load Batch Run*** (Ctrl L)

The *Fast File* format can speed up waveform viewing by over 100,000 times, so you can instantly load and display these Fast Files.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Start the Interactive Tool

If you have a license for Undertow Suite, you can start the Interactive part of Undertow, which include the source code, state machine, and schematic viewers, all of which are integrated with the Waveform viewer that you currently have open.

Add to Open Files

Opens a new file and keeps the window open.

Open

Opens a file (VCD, fast file, or Optimizing_tool output).

Closing Files

Table 7, “CLose Files Options” describes the two menu items that you can use to close files.

TABLE 7. CLose Files Options

Close Files	Opens a dialog box that you can use to unload any number of files that are currently loaded.
Close All Files	Closes all files, and clears the **Waveform** viewer, but keeps your Undertow session active.

Reread Dumpfiles

If you changed your design during the Undertow session, you can use this option to cause Undertow to reread the file, including the entire hierarchy Because design changes can eliminate signals, some of the signal names on the display might no longer exist.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Saving Undertow Sessions

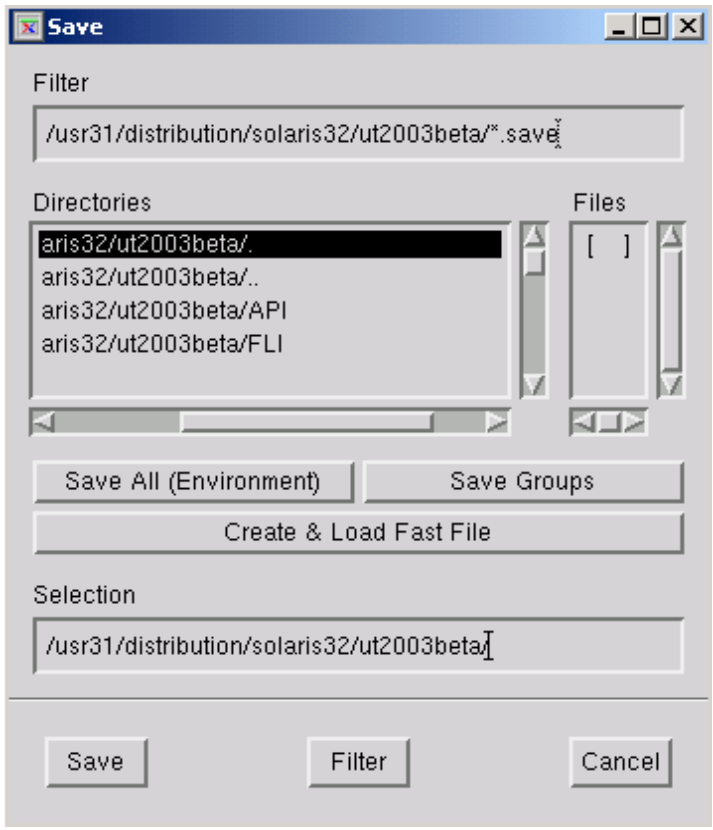


TABLE 8. Save Dialog

Feature	Description
Save All (Environment)	Saves the settings for the current display: signals, cursors, markers, groups, zoom, and so on.
Save Groups	Saves the groups that you defined in the current session, into a file.
Save	Saves the names of the signals that are currently displayed. You can use the signal list that this command generates for Groups (see Section , “Grouping Signals”), or with the Test Analyzer (see Section , “Test Analyzer”).

You define the file name when you save the file. After you save a file, you can use the ***File*** => ***Restore*** menu option to restore the file in future sessions of Undertow.

Restore

Restores previously-saved files (see Section , “Saving Undertow Sessions”).

- If the file contains a list of signals, Undertow displays those signals, but only if the signals exist in the current model.
- If the file holds grouped definitions, **Restore** adds those groups to the current set of groups.
- If you used the **Save All (Environment)** option to create the file, then the viewing environment is restored.

The Restore dialog box includes two selector windows. To **choose items**, you can either double click the **left** mouse button, or select the item and press the **Enter** (**Return**) key:

Start New Undertow

Use this menu option to start another session of Undertow. Although the new session runs as a separate session, it uses the same license as your original Undertow session; that is, you can run multiple copies of Undertow concurrently under the same license.

- If you close the primary session of Undertow, secondary sessions continue to run.
- You cannot select this option from a secondary session of Undertow. You can run more Undertow tools *only* from the primary (first) copy of Undertow that you started, but you can start multiple secondary sessions from the same primary session.

  The reason for this restriction is that when Test Analyzer compares signals, it compares the *primary* copy of Undertow against the window from which you started the Test Analyzer. Test Analyzer cannot compare two *secondary* windows.

This feature allows the user to start another complete copy of Undertow. This copy is just like the first copy, except that you can synchronize (**sync**) the new copy to the first copy of Undertow if you need to.

Sync

As described in the preceding section, you can use the ***File = > Start New Undertow*** menu option to run multiple sessions of Undertow concurrently, using the same license. You can run these sessions either independently or synchronized. In the synchronized mode, time scales can be to the same scale, and you can lock the scrolling of windows so they scroll together. Table 9, “Sync Undertow Options” describes the **Synchronize** options in the **File** menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 9. Sync Undertow Options

Sync Undertows Start-End	Synchronizes all Undertow windows to each other. When you move the horizontal scroll bar or the zoom controls in one window, you automatically perform the same action in the other Undertow window.
Sync Undertows Center.	• **On** lets you use different zoom factors in each window. • **Off** forces all Undertow windows to display the same portion of the simulation.
Sync Interactive	**On** synchronizes times between the Undertow T0 cursor and the Trace window in the Interactive_tool.

Printing Signals

Prints the signals that you selected in the current display, either on a line printer or to a file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

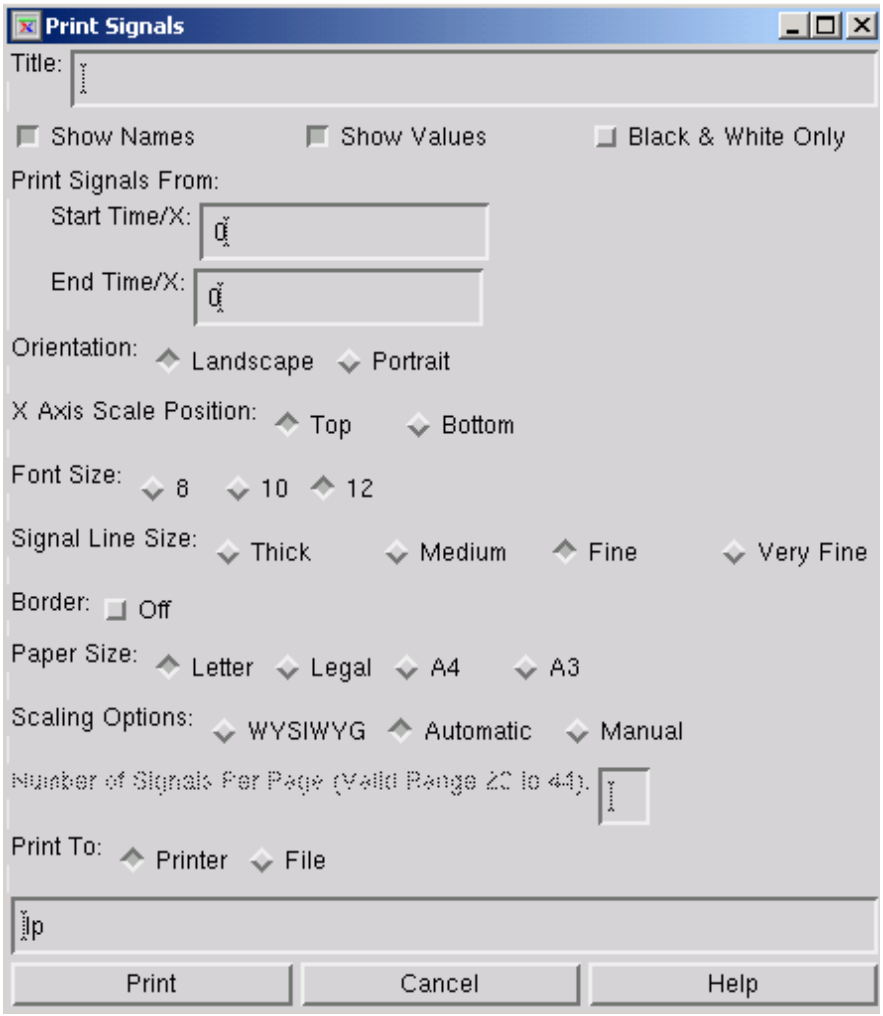


FIGURE 38. Print Dialog Box

The **Print Signals** dialog box includes the following input fields:

Scaling Options:

WYSWYG: Prints the screen as shown in the Undertow Display window

Automatic: Prints the screen, but spreads-out the signals vertically, to fill the print area. You can use this option to quickly print signals from simulations.

Manual: When printing, uses but uses the selected **Number of Signals Per Page** to determine the vertical size of each signal. Use this option if you want all signal sizes to be the same size in all diagrams—for example, in a manual.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 10. Print Dialog Box

Field	Description
Title	Printed output title.
Show Names	Displays signal names.
Show Values	Displays signal values.
Black & White Only	Displays signals in **Black and White**, rather than Gray Scale.
Start Time/End Time	You can enter integers in the range of the defined time limits (in current time units). Signals are printed in the scale shown in the current display window. If you increase the time scale, you can print the signals over several pages. You can then line up and tape together several pages of signals, providing one continuous waveform print. The position and length of the horizontal scroll bar defines the default start and end time for the print output.
Orientation	The default orientation is landscape. You can change it to portrait.
Font Size	**8**, **10** (default), or **12** points
Signal Line Size	Width of waveform lines. The default is **Fine**.
Border	Print with/without a border. The default is **with** a border.
Page Size	The default page size is **letter**.
Scale Options	Sets vertical scaling to **Automatic** (default) or **Manual**.
Number of Signals per Page	Specify the number of signals per page as an integer, from **27** (default) to **54**.
Print To	Either **Printer** (default) or **File**. • If the printer has postscript capability, the printer name should be **ps**. Otherwise, Undertow automatically selects an **lpr** printer. • If you select **File**, specify a file name instead of a printer name. • If a test comes out of the printer instead of graphics, set the **LPR_CALL** environment variable to the exact print command line, *without* the filename. (for example, set to **lp -d myprinter -o ps**).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Creating Files

TABLE 11. Files Create Submenu

Menu Item	Description
Create Fast File	Creates a new file that Undertow uses to display signals extremely fast. The program runs on either existing completed dump files, or VCD files while the Verilog simulator builds them. The Create Fast File option works in the background, so after you enable this function, you can continue to use other facilities of Undertow.
Create Reduced File	This feature has been removed.
Create Auxiliary File	This feature has been removed.

Format Menu

The **Format** menu controls how Undertow displays signals and buses. Table 12, “Format Menu” describes the options in this pulldown menu. Many of these operations work on multiple signals when selected. Ones complement and twos complement are for binary/hex values; Decimal ones/ twos complement are for decimal numbers.

TABLE 12. Format Menu

Menu Item	Description
Hexadecimal	Converts the value of the selected signal at the cursor into hexadecimal format.
Binary	Converts the value of the selected signal at the cursor into binary format.
Decimal	Converts the value of the selected signal at the cursor into decimal (base10) format.
Octal	Converts the value of the selected signal at the cursor into octal format.
ASCII	Converts each 8-byte character into its equivalent ASCII character.
Normal	If the value was converted to ones complement or twos complement, **normal** returns the value to the uncomplemented value.
Ones Compliment	Converts the value to the ones complement of the value.
Twos Compliment	Converts the value to the twos complement of the value. This is equivalent to the ones complement of the value with a one added to it. This is also equivalent to the binary inversion of this value.

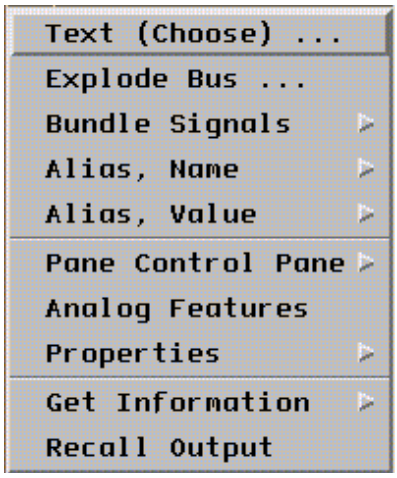
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 12. Format Menu (Continued)

Menu Item	Description
Logic	Changes an analog representation of a bus back to a logic (**0**,**1**,**X**, or **Z**) representation. To use this feature, select an analog representation of a bus, then select the ***Format => Logic*** menu option.
Analog	Creates an analog representation of a digital bus. To use this feature, select a digital bus, then select the ***Format =>Analog*** menu option.
Analog, Points	When you select analog signals, this option displays the actual points of the signal. Points display very small (one pixel), so you might need to turn off grids to see these points.
Analog Color	Opens a popup menu that you can use to choose the color for the selected signal(s).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Options Menu



Text (Choose)

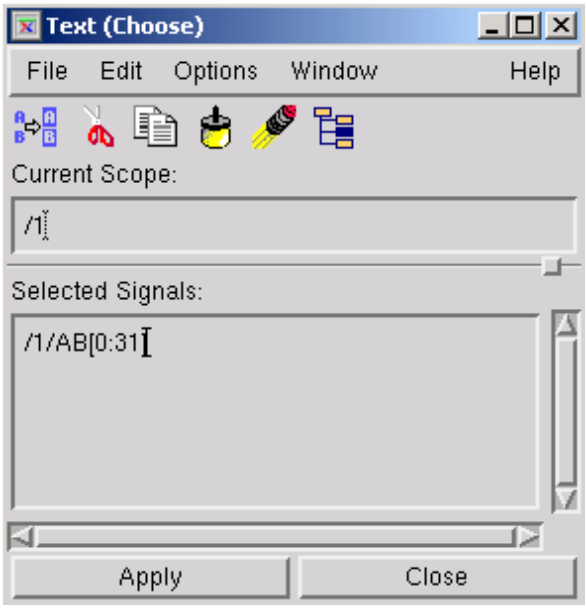


FIGURE 39. Text (Choose) Window

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

General Descriptions

This operation adds the user-speified signal to to the waveform display. You also use this feature to edit signal names that you either select and drag and drop from the signal display window or that you enter from the keyboard. This allows users to unbundle bus, flip bus and maniupulate signal display in different scope.

Menus

- **File**: **Open** - open a alias file

SaveAs / Save - save the data currently in the alias window.

- **Edit** - Select all/ Cut/ Paste/ Clear texts.
- **Window - Configure** - To configure the **Text Choose** Window, select the "**Configure** option from the Window pulldown menu.

Icon Menu

- **Select All/Cut/Copy/Paste/Clear**—Selects/Cut/Copy/Paste/Clear the whole contents of the **Selected Signals** text input area.
- **Set Scope**—set the scope specified by the Current Scope window. For example, if a user put /1/top and press **Set Scope**, Undertow will change the scope to the scope specifed in the window.

Current Scope Window

Use the **Current Scope** input field to modify the scope. This window allows user to specify the hierarchy/scope that user want to be in. By pressing "**Set Scope**" after specify the scope, Undertow will automatically change to the specifed scope.

Selected Signals Window

The **Selected Signals** scroll window lists signal names that will be concatenated to the scope. For example, if current scope is "/1/top" and the signal is "signal1", the complete signal name will be "/1/top/signal1".

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Window Control

- **Apply**—Adds the signals from the text window, to just above the selected signal name in the display. If you do not select a signal name, adds the new signals to the bottom of the signal list.
- **Close**—dismisses the Signal List window without doing anything.

Note on Signal Naming

- To let the signal name override the current scope, start the signal name with a **/** (slash).
- Signal names can contain ***** wild card characters (such as **c***, **ab***, and so on).
- You can *unbundle* signals that are grouped into a bus (for example, you can modify **Ab[31:0]** to **Ab[31:24]**).
- To reverse a bus, use the **Selecte**d **Signals** window:
  1. In the Waveform display window, select the bus to reverse.
  2. Drag and drop the selected bus signal to the **Selected Signals** window.
  3. Change the order in which the bus is specified in the Selected Signals window.
  4. Add the signal back to the waveform display window by pressing “Apply”.

     For example, you might change **AB[31:0]** to **AB[0:31]**.

Explode Bus

General Discription

This function display each bit of a digital signal bus.

1. Highlight the bus to expand.
2. Click **Explode Bus**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Bundle Signals

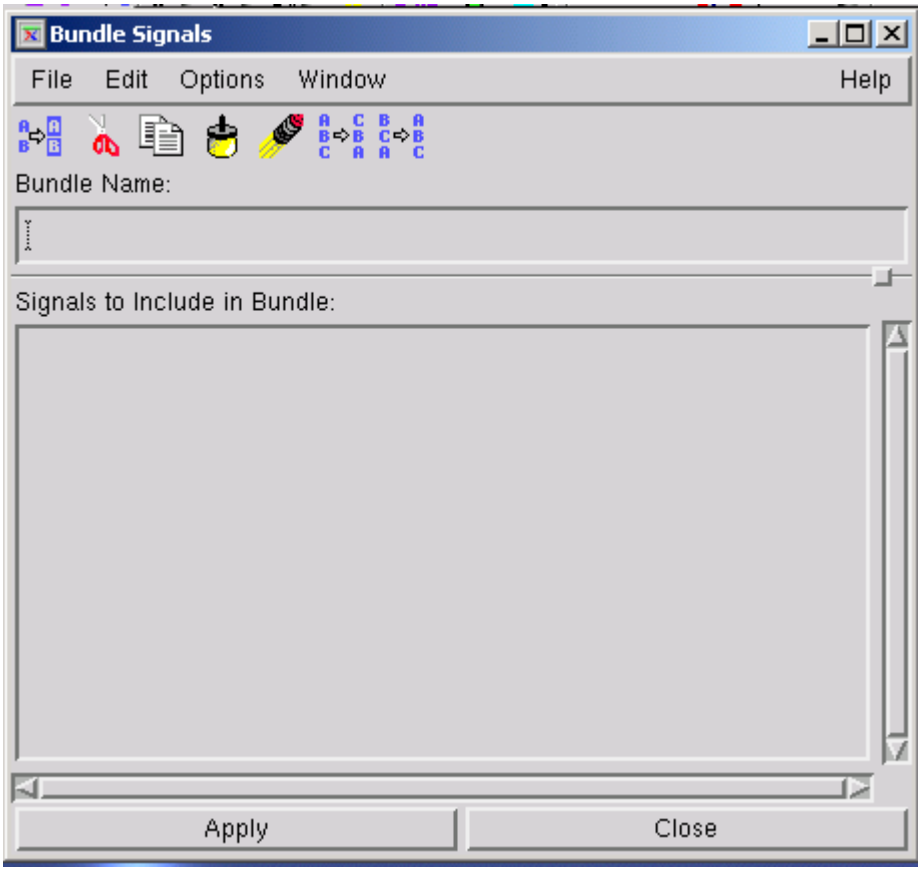


FIGURE 40. Bundle Signal Window

General Discription

Bundling creates/modifiy/list a vector out of bits in digital signals.

TABLE 13. Submenu for Bundle Signal

Submenu item	Description
New	Create or define the new Bundle Signal..
Modify Selected	Modify the selected bundled signal
List...	Opens a list of already Bundle Signal Window

Creating new Bundled Nets

Create a new bundled nets with selected nets

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Menus

- **File**: **Open** - open a alias file

  SaveAs / Save - save the data currently in the alias window.

- **Edit** - Select all/ Cut/ Paste/ Clear See 5.7.5.3 for details

Bundle NameWindow

This window allow user to specify the name to the new bundle. This name will be appeared in the wave form window as the new bundle is formed.

Signal to include in Bundle

User selects the signals to be bundled. User can either drag and drop or type the signal name.

Window Control

- Apply - apply the selected signal to the Bundle net name. After pressing apply, a new Bus should appeared in the waveform viewing window.
- Close - dismiss the Bundle Net Window without doing anything.

Modify an existing bundle

1. Select the bundle nets
2. **Options** ->**Bundle Signal** -> **Modify Selected**
3. Modify the bus in the **Bundle Signals** window

List the existing bundle

List the current bundles defined by user, Double click it to modify it.

Example for using Bundle Signals

To populate the bundle dialog box, as shown in Figure 42, “Creating a Bundle”

, do the following:

1. **Options** -> **Bundle Signals** -> **New**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1. Select the signals (**accelerate**, **b**, and **brake**).

2. Drag and drop these signals into the Bundle Signals area, called Signals to Include in Bundle.

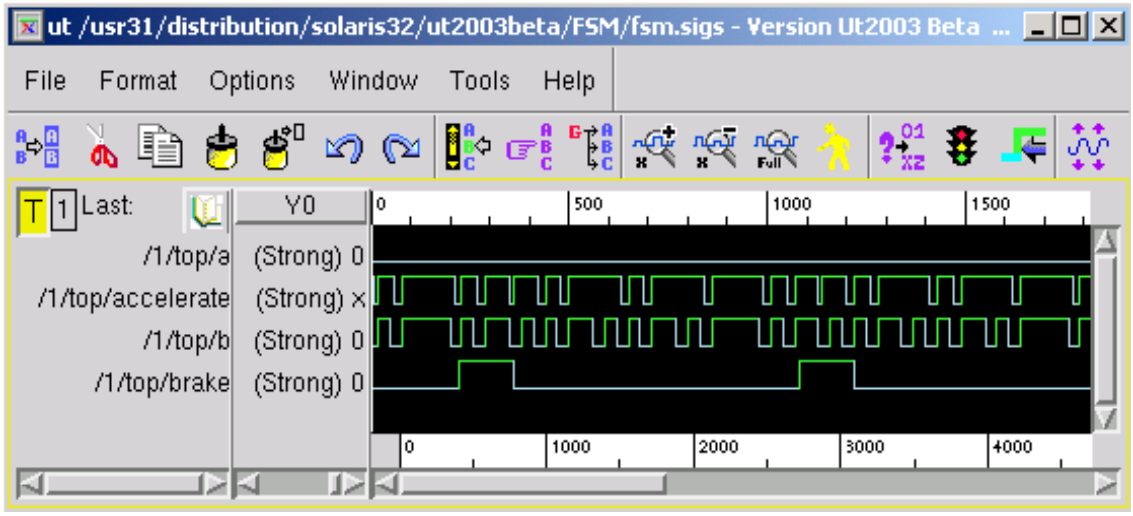


FIGURE 41. Waveforms Before Bundling

Note: To create a bundle without deleting the original signals, press and hold the Ctrl key while you use the middle mouse button to drag and drop. The drag and drop copies (rather than cuts) the signals.

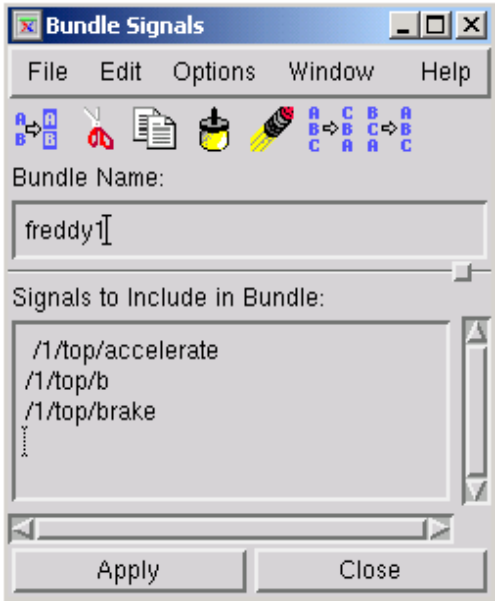


FIGURE 42. Creating a Bundle

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Click **Apply**.

   Figure 43, “Waveforms After Bundling” shows the results of the Bundle signals. The new bundle is called *Freddy1*, and is displayed below the *1/top/a* signal. Freddy1 is a 3-bit vector, denoted [2:0]. The numbers shown in the bundle (6, 7) represent the bit values for the selected time period.

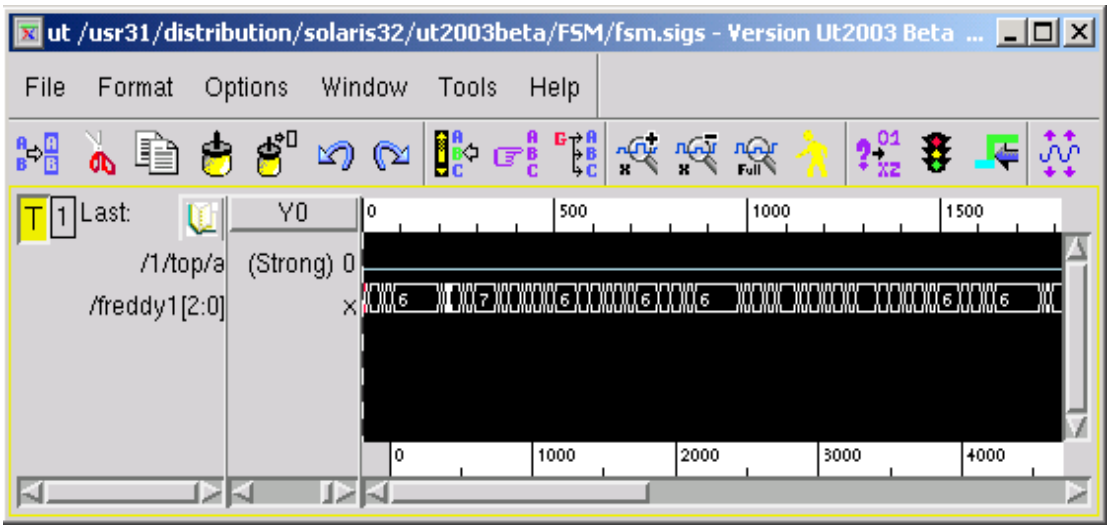


FIGURE 43. **Waveforms After Bundling**

Alias By Name

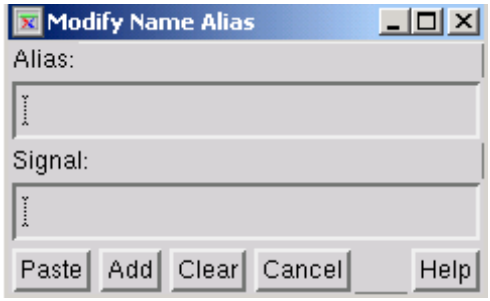


FIGURE 44. **Modify Name Alias**

General Descriptions

This operation allow users to define new name aliases or modify existing ones.

 Copyright Veritools 2004 All Rights Reserved

.

TABLE 14. Alias Name Submenu

Submenu item	Description
Name Alias	Opens it empty, ready to define the new alias.
Modify Selected	Opens it with the selected aliased signal ready for you to edit.
List	Opens a list of already aliased signal.

Setting Up New Name Alias

Allow users to modify the signal name to user-defined alias. User need to fill in the signal name in the signal box and filled in the corresponding alias name in the alias box. Upon pressing add, it will add this alias to the current signal. The alias will appear in the waveform window instead of the signal name.

TABLE 15. Alias Dialog Box

Feature	Description
Alias	Name to use for alias
Signal	List of signals to apply alias to.
Paste	Adds the signals in the buffer to the list of signals to alias.
Add	Adds the alias to the selected signal(s)
Clear	Clears the “Signals to Use the Aliases” input window.

Modifying existing alias

Select “Modifying Selected” bring s it up with the selected aliased signal ready to be edited. Selecting “List...” brings a list of already aliased signals, the user can click on a signal to bring up the modified name Alias window with the alias ready to be edited.

List

Select List will show all the existing alias that was defined by the user. Selecting the signal and double click it will allow user to modify this signal. To Modify the signal, please refer to the “modifying existing alias” section.

Opens a pull-down menu that you can use to either:

- Define new name aliases, or
- Modify existing aliases.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

You can use aliases for either of two purposes:

•To replace a text value with a name that represents that value, **Alias Values**, or

•To change the name of a signal, **Alias Names**

Alias By Value

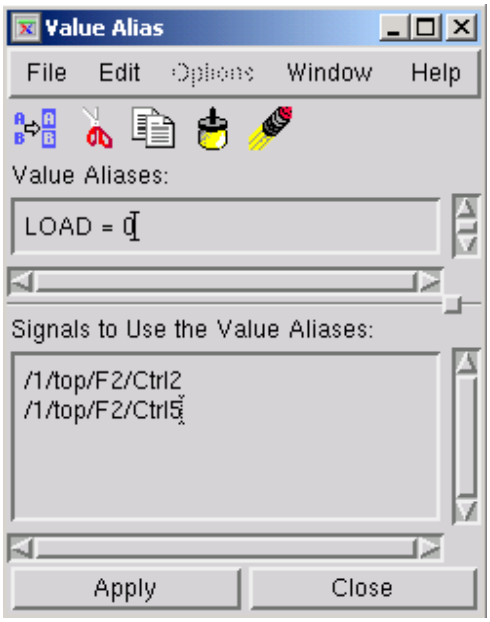


FIGURE 45. **Value Aliases**

General Descrptions

This Operation assigns/modifies alias to bus/signal values in the waveform viewer. The user can inputs a list of aliases and the signals to which the aliases should apply. After applying the aliases, the alias will appear in the waveform viewing window instead of the original numerical value.

Menus

- **File**: **Open** - open a alias file

SaveAs / Save - save the data currently in the alias window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Edit** - Select all/ Cut/ Paste/ Clear See 5.7.5.3 for details
- **Window - Configure** - To configure the **Value Alias** Window, select the "**Configure** option from the Window pulldown menu.

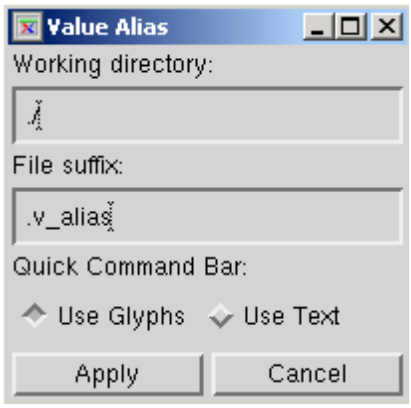


FIGURE 46. **Configuring Value Window**

TABLE 16. **Alias Values Configure Window**

Menu Item	Description
Working Dir	Directory where alias file resides.
File Suffix	Filter for Alias files.
Quick Command Bar	Choose between pictures or text.

Icon

- **Select All**—Selects the whole contents of the **Selected Signals** text input area.
- **Cut**—Cuts the selected text from the **Selected Signals** input window.
- **Copy**—Copies the selected text from the **Selected Signals** input window to the current cursor position.
- **Paste**—Pastes the signal names or signal groups (which you previously cut or copied) into the Text Window. You can modify signal names on the Text Window, to unbundle a bus. You can type signal names directly into the Text Window; these signal names can include wild card characters (such as *****).
- **Clear**—clears the **Selected Signals** input windows.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Value Aliases Window

Value alias window allow user to assign aliases to bus or scaler values (syntax: *alias* = *value*)

Example:

LOAD = 001

MOVE = 00B

Note: units for values must be at radix of the signal

Signal to use Value Alias

This window holds the names of the signals to be assigned to the alias values. The signal can be dragged and dropped into this window or typed into this window. If a signal is selected prior to pressing **Alias By Value** menu item, the signal will be placed in this window when the widow is opened.

Window Control

- **Apply** - assigns aliases to bus values
- **Close** - closes window without assigning aliases if apply is not clicked

Example

1. Select the bus by clicking on the bus at the signal naming area in the wavefor viewer.
2. Select the **Alias, Value** by **Options** -> **Alias, Value** -> **New**.
3. The selected bus will automatically be placed in the "**Signals to Use the Value Alias"** window contained in the Value Alias window
4. **Left** click on the upper window named **Value Aliases**.
5. Enter the aliases to be assigned to the bus values. such as START = 1FF.
6. **Left** click on the **Apply** to add the alias to the waveform viewer display.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

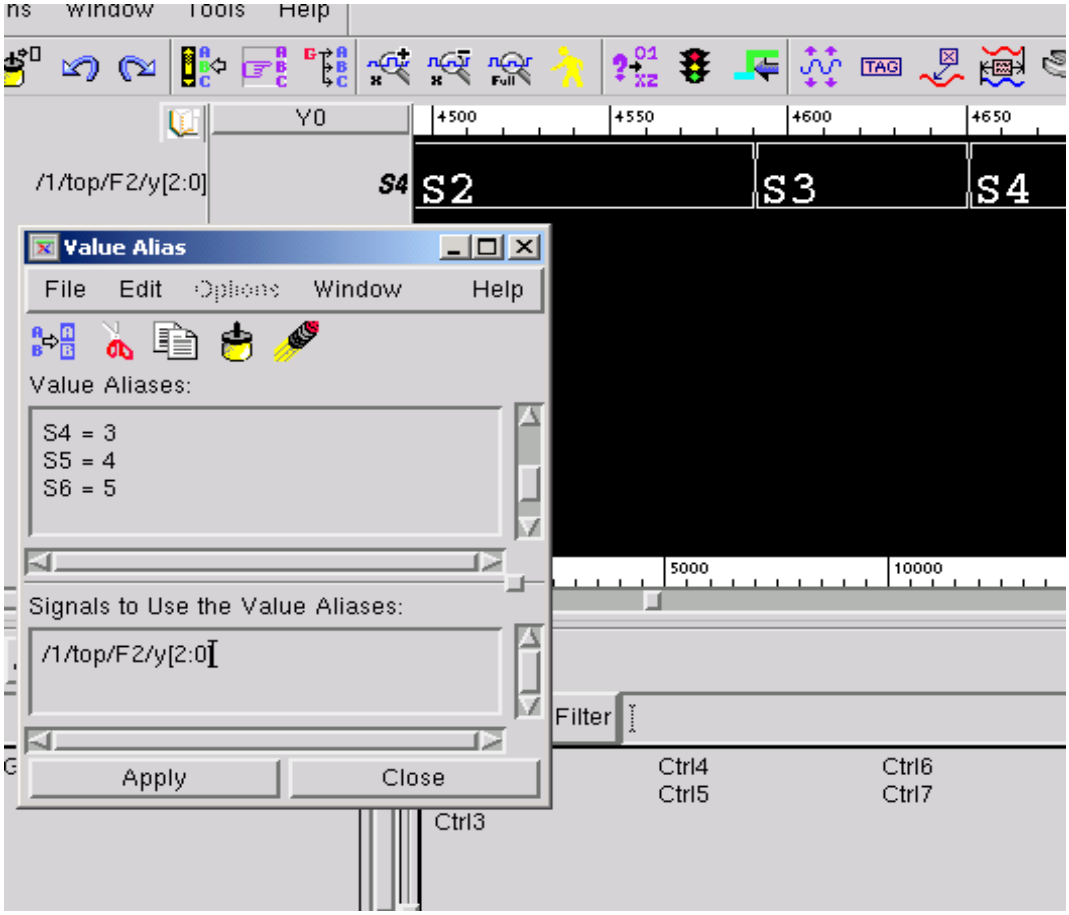


FIGURE 47. Example of using Value Alias

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Pane Controller

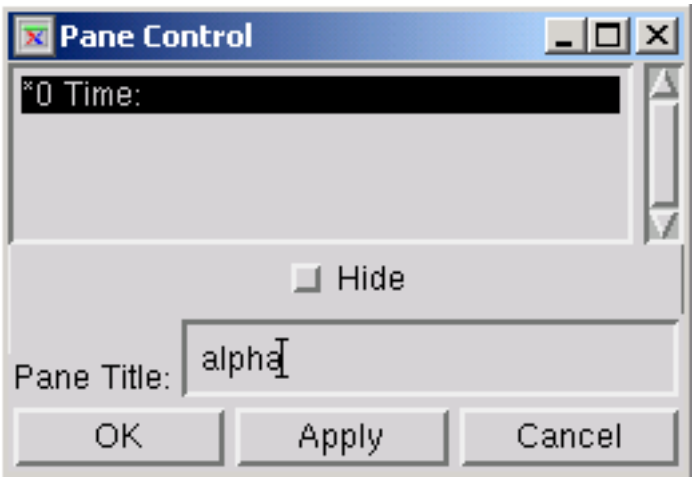


FIGURE 48. **Pane Controller Window**

TABLE 17. **Pane Control Dialog Fields**

Field	Description
List	List of open panels.
Hide	Hides a panel if more than one is open.
Panel Title	Names a selected panel.

Use **Pane Control** to make any pane visible, or to hide any pane (as defined in the Pane Control window).

Pane Control Zoom Time

Use **Pane Zoom** to set the pane defaults for the pane that is in focus. You can use this window to set both the bottom or minimum values for each overlay area, and the top or maximum value for each overlay in the focused pane for **Time/X**, **Voltage**, **Current**, and **Power**. You can also set the units for each of these parameters.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

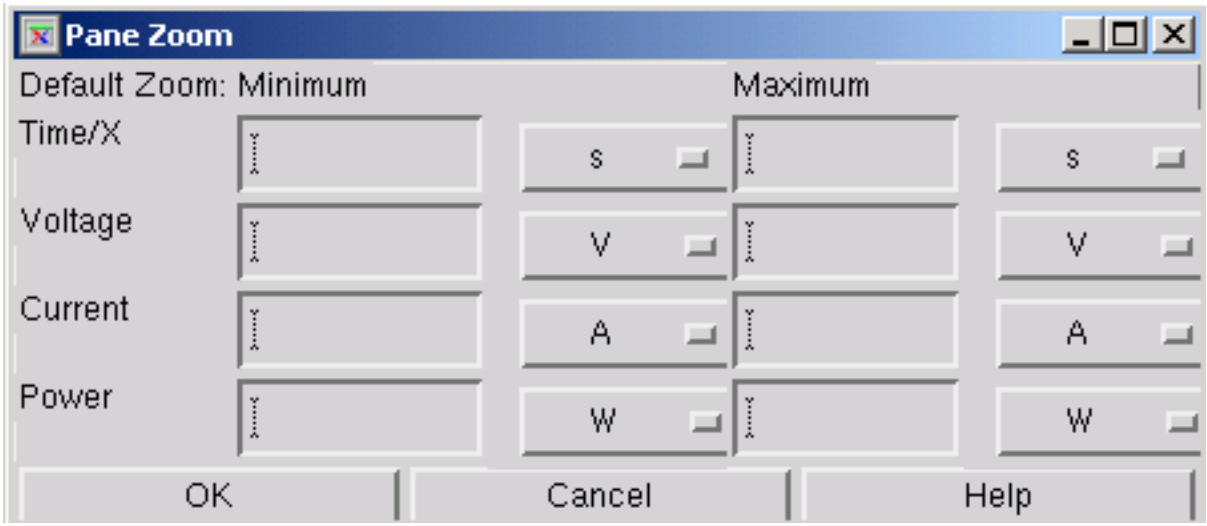


FIGURE 49. **Pane Zoom Window**

Properties

This section describes properties that you can set from the Undertow ***Options*** => ***Properties*** menu item.

Overlay Files

Opens the Overlay File window, which includes the following:

- On the left side, the File name area, then the alias for this file (also called the Hierarchy Mode).
- On the right side, the file designator or Option. You can select the following file designators: **<none>**, **Overlay 1** (the Base file), **Overlay 2**, or **Overlay 3**.

The Waveform *overlay* panel refers to a single vertical position in the waveform window., This panel has the signal name/names and the signal value/values at the left, and the waveform at the right.

The overlay panel can display one waveform, or many waveforms that are overlaid. It has a border above and below, which you can change using the **Waveform Spacing** option.

1. To show all overlays, click the **Select All** icon.

   Undertow outlines each overlay with a red border in the waveform display area, and with a light grey boarder in the signal name area of the waveform display window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select the file designator for any file loaded.

   If the file has no designator, Undertow indicates “none”.

3. After you set the designators, move to the Choose window and select ***Options*** => ***Properties*** => ***Overlay Files***.

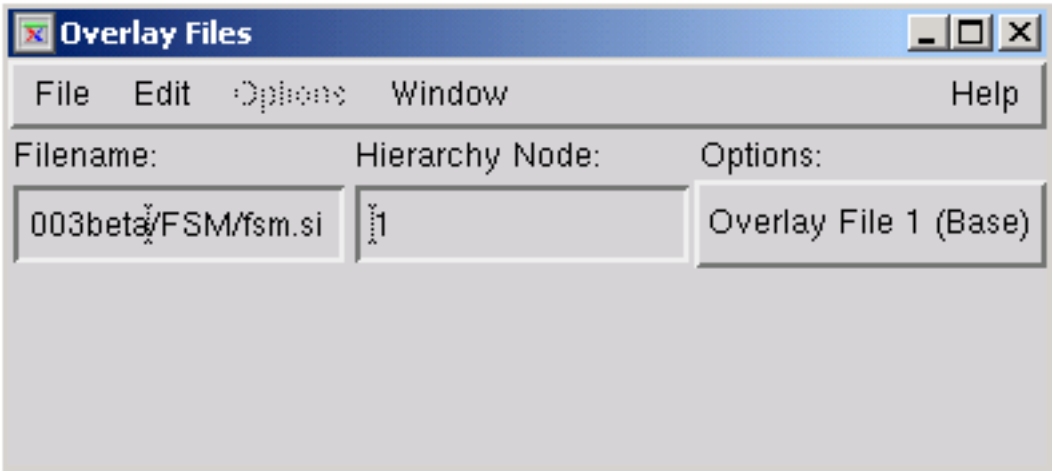


FIGURE 50. **Properties Overlay Dialog Box**

Setting Fonts

To open the fonts dialog box (Figure 51, “Options => Properties => Fonts Dialog Box”), select **Options=> Properties => Fonts** from the menus.

 Copyright Veritools 2004 All Rights Reserved

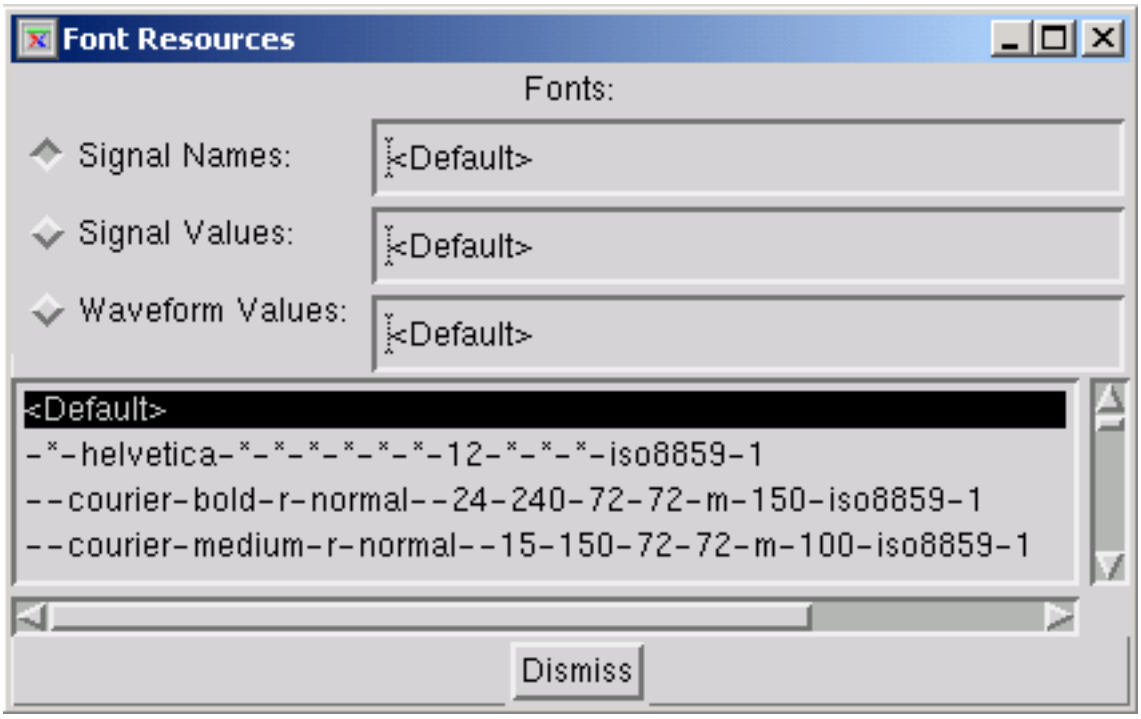


FIGURE 51. **Options => Properties => Fonts Dialog Box**

1. To change a font, select (toggle) the labels of interest: signal names, signal value, or waveform values.

2. From the list of available fonts select the desired font.

Getting Information

Use the **Get Information** submenu to view and extract the data for selected signals. This information is displayed in a text window, as shown in Figure 52, “Get Information Dump Output”.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

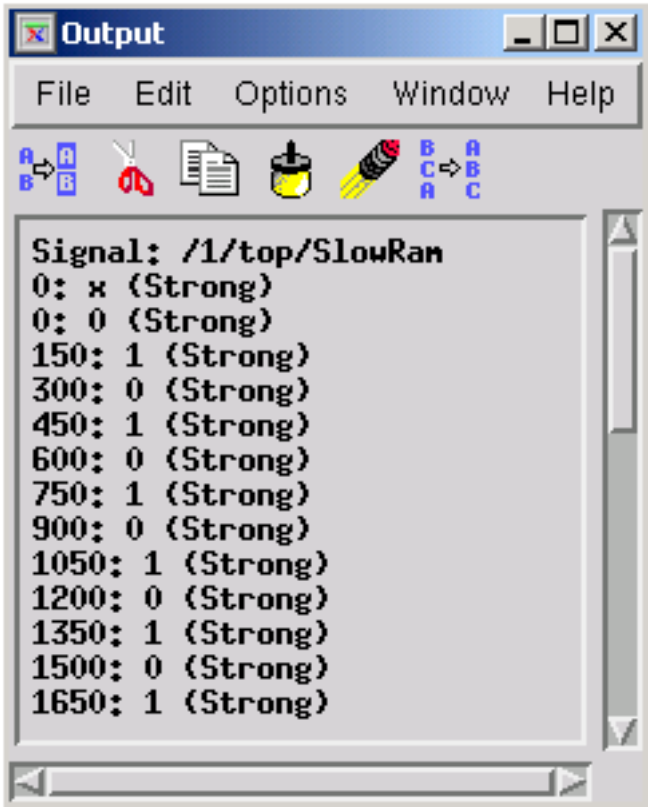


FIGURE 52. Get Information Dump Output

From the **File** menu in the **Output** dialog, you can save this information as a ASCII (text) file.

To Dump Signals in a Hex format, do the following:

1. Select Signals to where you want to display information.
2. Click ***Options*** => ***Bundle Signals*** => ***New***.
3. Drag and drop the selected signals to the **Bundle** window.
4. Click **Apply** and add this bundle to the signal list.
5. Select this bundle, and then select ***Options*** => ***Get Information Dump Signals***.

Undertow then outputs the information for this set of signals in Hex format.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 18. Options Dump Submenu

Menu Item	Description
Dump Signals	Select the signal that you want to see, and click the left mouse button on **Dump Signal**. The text shows the time and signal values for the selected signal.
Dump Signals, Tabular	Select the signals that you want to see, and click the left mouse button on **Dump Signals, Tabular**. The text shows the time and signal values for the selected signals, in a tabular form.
Dump Names	Displays the full hierarchical name for the selected signals, and the type of signal, such as analog, Compact Bus (DI[31:0]), Scalar Signal, SplitBus(DI0, DI1, etc.), and so on.
Dump Bundles	Displays the name of the selected Bundle, and the signals in that bundle.
Dump Value Aliases	Displays the value aliases for the selected signal.

Recall Output

This is a context-based output window. It displays output from various Undertow functions. The format of the output depends on which function dumps the information to this window. For example:

- Select ***Options => Get Information Dump Signals*** to dump the data onto this window.
- Select ***Dump Signals, Tabular*** to dump the data in a tabular format.

When you execute a script from the Script analyzer, the output of this script is put on this window, and so on.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Window Menu

Configure Dialog

When you click the **Configuration** button (shown at the left), the **Configuration** Window opens (see Figure 74, “Analog Configure General Dialog”).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

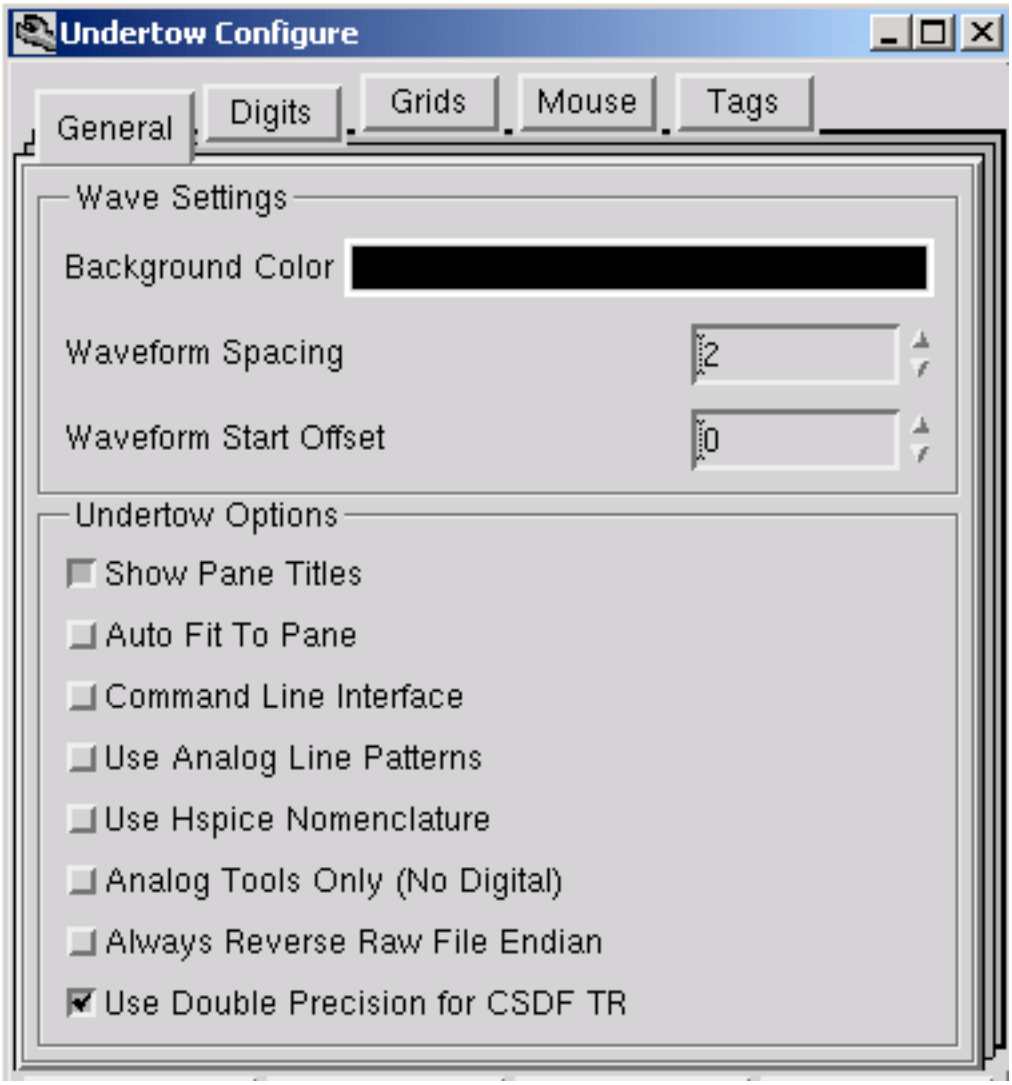


FIGURE 53. **Configuration General Dialog**

TABLE 19. **Configure General Dialog Fields**

Attribute	Description
Waveform Background Color	Pops up a list of colors that you can select for the waveform display area. To change the waveform background color, select a color from the list, and click **Apply**. Standard colors are *black* and *grey35*.
Add Color To List	Use this option to add a color to the **Waveform Background Color** popup list. You can add any valid color entry found in the **rgb.txt** file, or any valid hex value (such as *#7799aa*).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 19. Configure General Dialog Fields (Continued)

Attribute	Description
Waveform Spacing	Sets the number of pixels that separate signals, or between waveform overlay panels. The default is 2 pixels. For example, 20 pixels provides about 1/4 inch between waveform overlay panels.
Waveform Starting Offset	Specifies the displacement (in pixels) along the x axis, for the starting point of signals. Use this option to set blank space in front of the displayed waveform. An offset of 20 provides about 1/2 blank space at full zoom; this space increases with each zoom-in operation.
Show Pane Titles	If you select this toggle, Undertow displays a title above each pane, if you created a title for that pane.
Right Click Pops Up Wave Window Menu	• If you toggle this option **on**, you can click the right mouse button in the **Waveform** viewer to pop up a dialog box for zooming and other operations. • If you toggle this option **off**, when you click the right mouse button, a zoom-in operation starts.
Auto Fit To Pane	If you toggle this button **on,** Undertow scales all signals to fit completely in the **Waveform** viewer. Selected signal(s) fill the entire pane.
Command Line Interface	If you toggled this option **on**, a one-line text window appears at the bottom of the Undertow window. You can use this text window to type commands, instead of using the mouse and dialog boxes.
Use Analog Line Patterns	Displays and prints 8 different line patterns, for selected analog signals in the same overlay.
Use HSpice Nomenclature	*Use HSpice Nomenclature* has no immediate effect. It takes effect the next time that you open a file or execute **Reread Dumpfile**. This option applies only to `.ac0` files, and only if there are hierarchical characters (“.”) in the signal names. These become modules if they do *not* use Hspice nomenclature.
Analog Tools Only (No Digital)	Moves all non-analog tools from the Tools menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

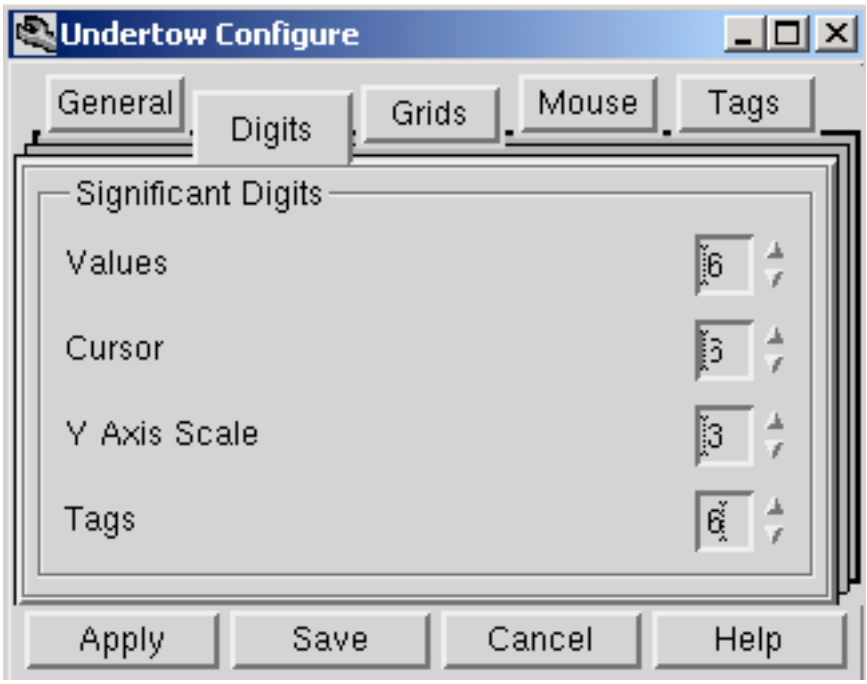


FIGURE 54. Configure Digits Dialog

TABLE 20. Configure Digits Dialog Fields Scales

Attribute	Description
Values Significant Digits	Sets the number of digits that appear in the **Value** area of the **Waveform** display. The **Value** area is located between the signal names and the waveforms.
Cursors Significant Digits	Sets the number of digits that the cursors use. These digits appear at top of the Waveform window, or on cross hairs if you selected the *cross hair value* display.
Y-Axis Scale Significant Digits	Sets the number of digits shown on the Y (Vertical) Scales.
Tag Significant Digits	Specifies the number of significant digits to show for the Tag Values.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 55. Analog Configure Grids Dialog

TABLE 21. Configure Grids Dialog Fields

Attribute	Description
Analog Grids	• If you toggle this option *off*, the **Waveform** viewer uses the default grid spacing. • If you toggle this option *on*, the grid spacing is set as you specified in **Minimum pixels per horizontal grid** and **Minimum pixels per vertical grid**. You can also display the left and right scales.
Minimum pixels per horizontal grid	This option specifies minimum pixels per *horizontal* grid.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 21. Configure Grids Dialog Fields (Continued)

Attribute	Description
Minimum pixels per vertical grid	This option specifies minimum pixels per *vertical* grid.
Show left scale	Flags for showing the left scale.
Show right scale	Flags for showing the right scale.
Minimum pixels per major scale division	This option specifies the minimum pixels per *major* scale division.
Minimum pixels per minor scale division	This option specifies the minimum pixels per *minor* scale division.

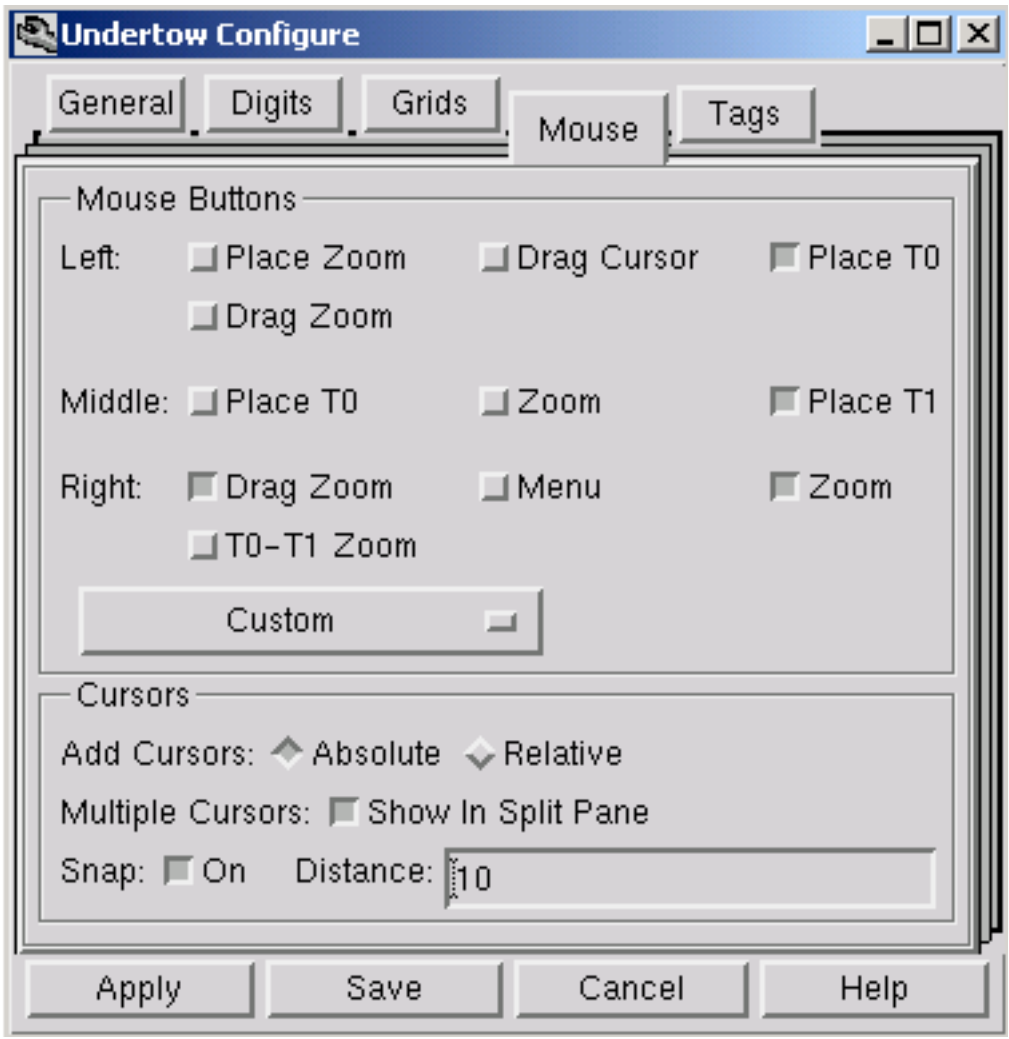


FIGURE 56. Analog Configure Mouse Dialog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 22. Configure Mouse Fields

Attribute	Description
Left Mouse	Can be set to provide the following operations; *Place Zoom*, allows user to place a dashed line for the left and then the right zoom boundaries, *Drag Cursor*, after pressing Add Cursor, this allows the user to drag and place this cursor *Place T0*, allows the user to place the T0 cursor when pressed
Middle Mouse	Can be set to provide the following operations; *Place T0*, places the T0 cursor when the middle button is pressed *Zoom*, does a zoom operation when the middle mouse button is pressed *Place T1*, places the T1 cursor when the middle button is pressed
Right Mouse	Can be set to provide the following operations; *Drag Zoom*, press down this mouse button, slide and zoom over slide area when released *Menu*, if pressed down and held a menu comes up with window operations that can be preformed *Zoom*, does a zoom when right mouse button is pressed *T0-T1 Zoom*, zooms between T0 and T1 when right mouse button is pressed

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

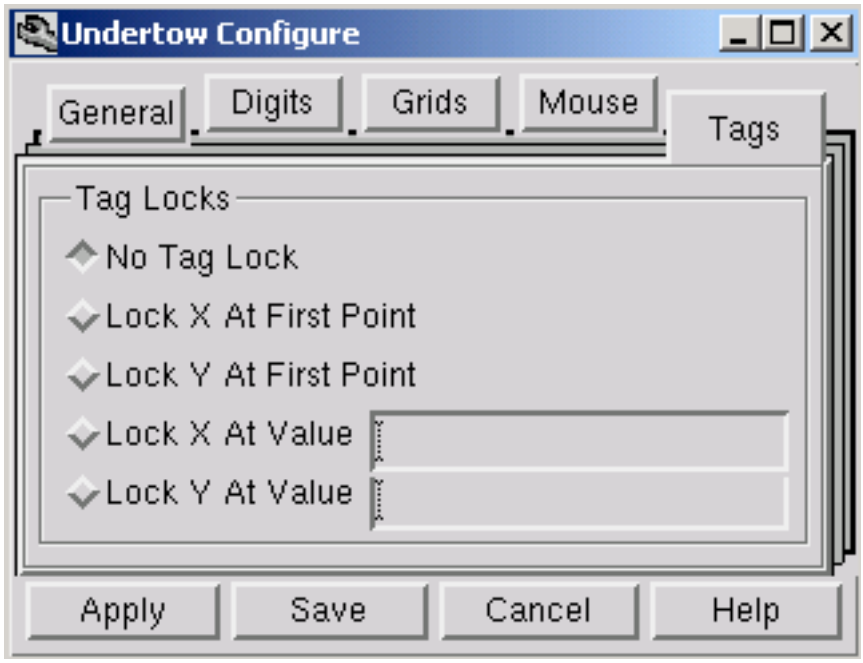


FIGURE 57. Configure Tags Dialog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 23. Configure General Tags Fields (Sheet 1 of 4)

Attribute	Description
No Tag Lock	Use this option to place tags on any point on any signal. • If you select a signal in the selected overlay, the cursor target appears only on the selected signal or signals. • If you do not select a signal in the selected overlay, the cursor target appears on all signals in this overlay. *Note: This cursor target action is the same for this mode, and for all modes listed in this table.* 1. Click on the **Measures Tag** or **Value Tag** icon. 2. When the cursor target is visible, click the left mouse button. The first anchor target is set. 3. Click the left mouse button to set the second anchor, if required. The tag automatically appears between the anchor cursors. 4. Repeat this process as often as you need to. 5. To end the process, click the **Close** button on the **Tag** popup dialog box.
Lock X At First X Point	You can place tag anchor points for the **X** axis. The first anchor point that you set, sets all other anchor points to this same value. *Note: The anchor target displays only when the cursor arrow is over the value for the **X** first point for any signal, indicating that you can place the tag anchor at this point.* If you set this mode while the **Measure** or **Value** tag is toggled on, then the cursor tag is visible *only* until you click the left mouse button, which places the first tag anchor point. After that, the cursor target is visible only at the same **Y** value on different signals in this overlay. If a cursor target appears over the same **X** value on another signal, you can use the left mouse button to place and set the second anchor point. The tag appears between the two anchor points.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 23. Configure General Tags Fields (Sheet 2 of 4)

Attribute	Description
Lock Y At First Y Point	You can place tag anchor points for the **Y** axis. The first anchor point that you set, sets all other anchor points to this same value. *Note: The anchor target displays only when the cursor arrow is over the value for the **Y** first point for any signal, indicating that you can place the tag anchor at this point.* If you set this mode while the **Measure** or **Value** tag is toggled on, then the cursor tag is visible *only* until you click the left mouse button. After that, the cursor target is visible only at the same **Y** value on different signals in this overlay. If a cursor target appears over the same **Y** value on another signal, you can use the left mouse button to place and set the second anchor point. The tag appears between the two anchor points.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 23. Configure General Tags Fields (Sheet 3 of 4)

Attribute	Description
Lock X At Value	When you place the tag anchor points, this option sets all anchor points to *only* this **X** value. *Note: The anchor target displays only when the cursor arrow is over the value for the **X** first point for any signal, indicating that you can place the tag anchor at this point* 1. If you sets an **X** value in the text area, set this mode, and toggle on the **Measure** or **Value** tag, then the cursor tag is visible whenever the cursor target is over this value on a signal. 2. If the cursor target is visible, click the left mouse button to place the first tag anchor point. The cursor target is visible *only* on the same **X** values on any signal. 3. You can set the second anchor point *only* when the cursor target is visible on the signal. The tag appears between the two anchor points. *Note: Set the **X** value first, and then set this mode. If you set the mode **before** you set the value, then the value does not take effect until you do one of the following:* • Click Apply, or • Click OK, or • Toggle this mode *on* again.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 23. Configure General Tags Fields (Sheet 4 of 4)

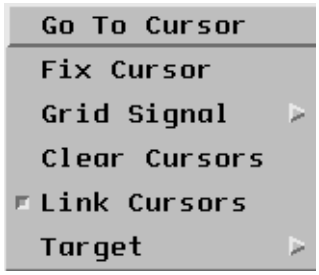
Attribute	Description
Lock Y At Value	When you place the tag anchor points, this option sets all anchor points to *only* this **Y** value. *Note: The anchor target displays only when the cursor arrow is over the value for the **Y** first point for any signal, indicating that you can place the tag anchor at this point* 1. If you sets a **Y** value in the text area, set this mode, and toggle on the **Measure** or **Value** tag, then the cursor tag is visible whenever the cursor target is over this value on a signal. 2. If the cursor target is visible, click the left mouse button to place the first tag anchor point. The cursor target is visible *only* on the same **Y** values on any signal. 3. You can set the second anchor point *only* when the cursor target is visible on the signal. The tag appears between the two anchor points. *Note: Set the **Y** value first, and then set this mode. If you set the mode **before** you set the value, then the value does not take effect until you do one of the following:* • Click Apply, or • Click OK, or • Toggle this mode *on* again. You must set the units for the **Y** value—for example, **1.75V**, or **2.330ua**. To determine the required units for the text value, press the middle mouse button on the desired point. *Note: If you do not set the units, this feature cannot find the value that you are looking for.*

Navigator

This option lists windows that are open. If a window is obscured by other windows, to raise it to the top of the window stack, select it from this list.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Cursors



Go To Cursor

Moves display to the position of the T0 cursor.

Fix Cursor

Fixes the cursors on the screen. The cursor does not move from its position, regardless of zooming and scrolling. The cursor does not have a fixed time position.

Grid Signal

LE (Leading Edge) = Rising Edge, **TE** (Trailing Edge) = Falling Edge

TABLE 24. Window Grid Signal Submenu

Menu Item	Description
Grid Signal LE	This option adds a dashed cursor to the leading edges of the selected signal. This operation is particularly helpful for checking the relationship of clocks to other signals that are anywhere in the Waveform display window.
Grid Signal TE	This option adds a dashed cursor to the trailing edges of the selected signal. This operation is particularly helpful for checking the relationship of clocks to other signals that are anywhere in the waveform display window.

Clear Cursors

Clears all cursors that you previously set.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Link Cursors

Use this option to link the horizontal and vertical cursors for Analog signals. The cursors cross where the mouse pointer is located, and follow as you move the mouse.

Target

TABLE 25. Target Cursor Submenu

Menu Item	Description
Small Cross Hair	Changes the cross hair cursor to a small cross hair.
Cross Hair	Changes the *arrow* mouse cursor to a cross hair cursor
Vertical Line	Changes the mouse cursor to a vertical line.
Default Cursor	Sets the cursor back to the default arrow cursor
Show X (Time)	Displays the time (x-axis value) on the mouse cursor's North East quadrant.
Show Y (Value)	Displays the value (y axis) on the mouse cursor's North East quadrant.
Show X,Y	Displays both X and Y values
Target Color	

Lock Cursors

TABLE 26. Window Lock Cursor Submenu

Menu Item	Description
Lock Cursor 0	Lock Cursor 0 locks cursor 0 to the selected signal. After you lock the cursor, you can use the middle mouse button to move cursor 0, while the cursor target precisely follows the signal.
Lock Cursor 1	Lock Cursor 1 locks cursor 1 to the selected signal. After you lock the cursor, you can use the middle mouse button and control key to move cursor 0, while the cursor target precisely follows the signal.
Lock Cursor 2	Lock Cursor 2 locks cursor 2 to the selected signal. After you lock the cursor, you can use the middle mouse button and shift key to move cursor 0, while the cursor target precisely follows the signal.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Annotate Waveform

Table 27, “Annotate Options” describes the three methods that you can use to annotate the **Waveform** viewer with time values.

TABLE 27. Annotate Options

Time Change	Displays the time changes of a signal. Places the annotation just to the left of the signal change.
Time Difference	Shows the time difference between two signals edges. Places the annotation in the middle of the signal transition that you are measuring.
Value Changed	Shows the value of a signal. Places the annotation just to the right of the transition.

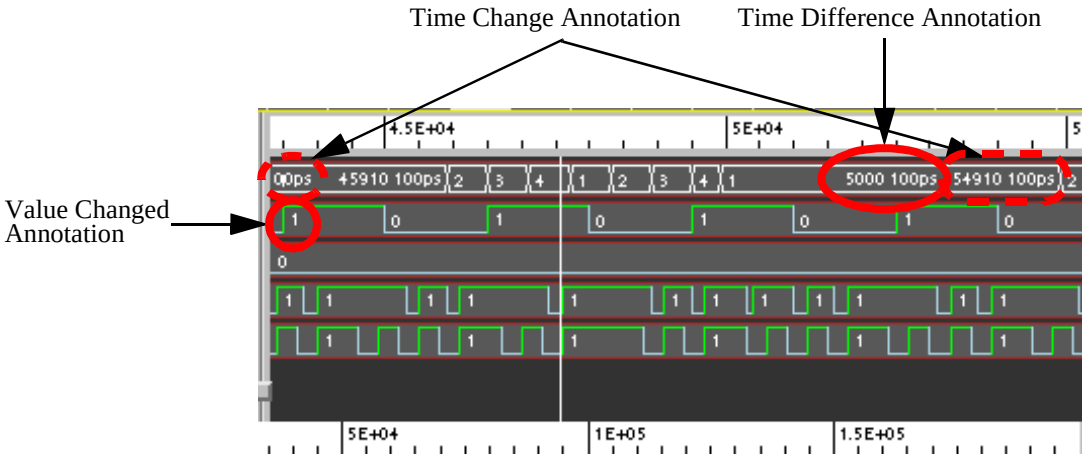


FIGURE 58. Time Annotation of Waveforms

Tags

You can use tags to annotate your **Waveform** viewer with text icons, as shown in Figure 59, “Tags”.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

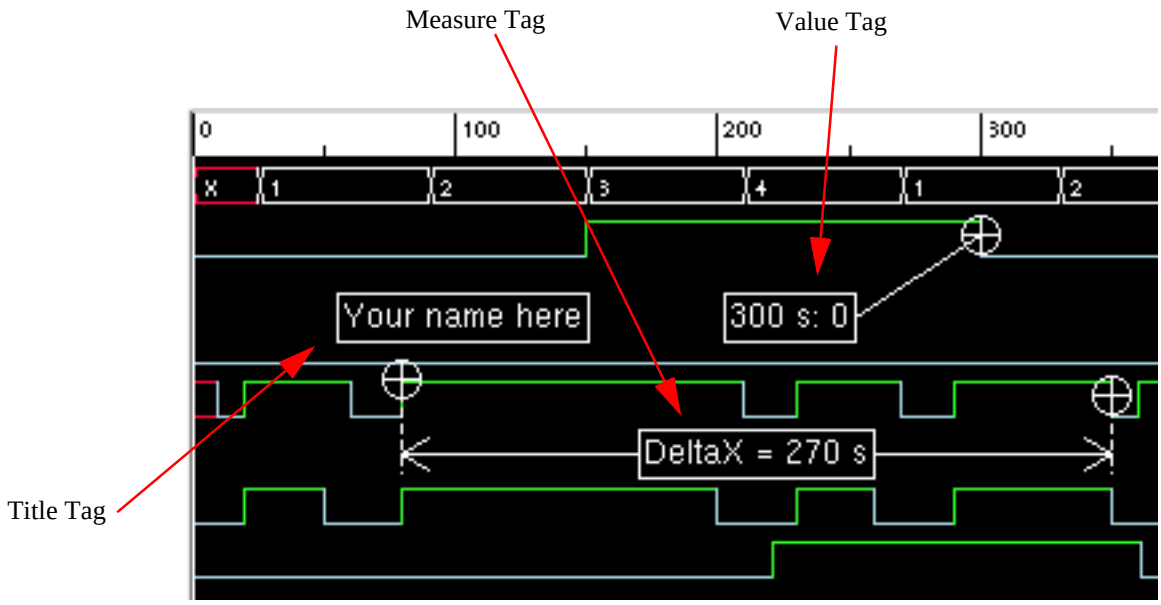


FIGURE 59. Tags

Title Tags

Title tags annotate your Waveform viewer with text.

1. To place a title tag, either select from the tool bar or as a menu option.

   The tile tag popup opens, as shown in Figure 60, “Title Tag”.

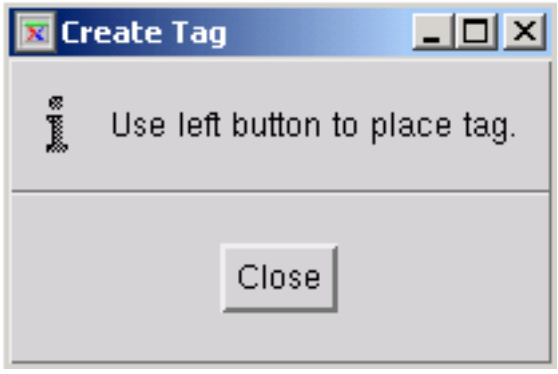


FIGURE 60. Title Tag

2. To place one or more tags, click on the **Waveform** viewer at the desired location for the tag.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

You can place as many tags as you need.

3. To finish the tag mode, click the **Close** button in the dialog box.

   By default the text of the tag is *Tag*.

4. To change this text at any time, click the right mouse button on the tag that you want to change.

   A menu pops up with the **Set Tag Text** option.

5. To move a tag, use the left mouse button to drag and drop the tag.

   Note: You must close the Create Tag dialog (Step 3) before you can move a tag.

6. To remove a tag, right click on the tag to open a popup dialog box, then select the **Remove Tag** menu option.

Value Tags

If you toggle on the **Value Tag**, then each time you click the **left** mouse button, you place a value point. A tag also appears, attached to each of the these value points.

- This value tag is a line at the value point, pointing to the tag. The tag contains the time value at the value tag point.
- **Tag Anchors** sets the tag between the T0, and T1 tag anchors. You can set these cursors at any point, on any signal. You can use all of the features in Undertow to set the cursors, the **GoTo** icon, Perl, and so on. After you set the T0 and the T1 cursors exactly where you want them, click **Tag Anchors T0/T1** to set the tag between these anchors.
- If **Snap** is on, then click the **left** mouse button to snap the cursor and the tag point to the nearest edge of the signal, in the horizontal plane of the cursor.
- Use the left mouse button to move the value tag. You can also use the left mouse button to move (drag and drop) the tag attachment point to a new location on the waveform display.
- You can continue to add tags until you either click off the **Value Tag,** or select **Close** on the pop up window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

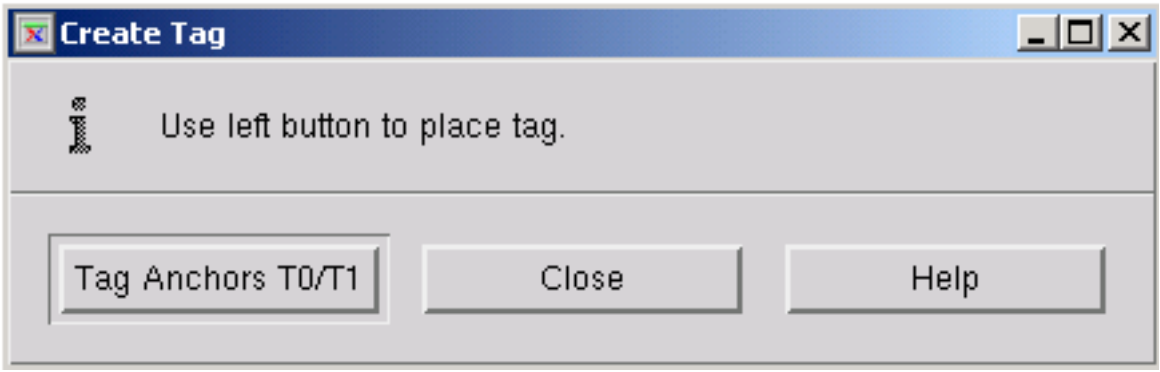


FIGURE 61. **Value Tag Dialog Box**

Measure Tags

1. Toggle on the **Measure Tag**,.
2. Click the **left** mouse button, to place the left measure point.
3. Click the **left** mouse button again, to place the right measure point.

   Undertow adds a measure tag between these points, with the delta time in this tag. The measure tag is a line, with arrows pointing left and right to the measure points. The delta time value is in the middle of this line.

- If **Snap** is on, then click the **left** mouse button to snap the cursor and the tag point to the nearest edge of the signal, in the horizontal plane of the cursor.
- Use the left mouse button to move the **Measure** tag. You can also use the left mouse button to move (drag and drop) the tag points to a new location on the waveform display.
- You can continue to add tags until you either click off the **Measure Tag,** or select **Close** on the pop up window.

Fit To Pane

This option enlarges the vertical size of the selected signal(s), to fit the pane of the viewer. To turn this feature off and display the signals in their normal context, press either the page up or page down key on your keyboard, or use the arrows at the top and bottom of the vertical scroll bar.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

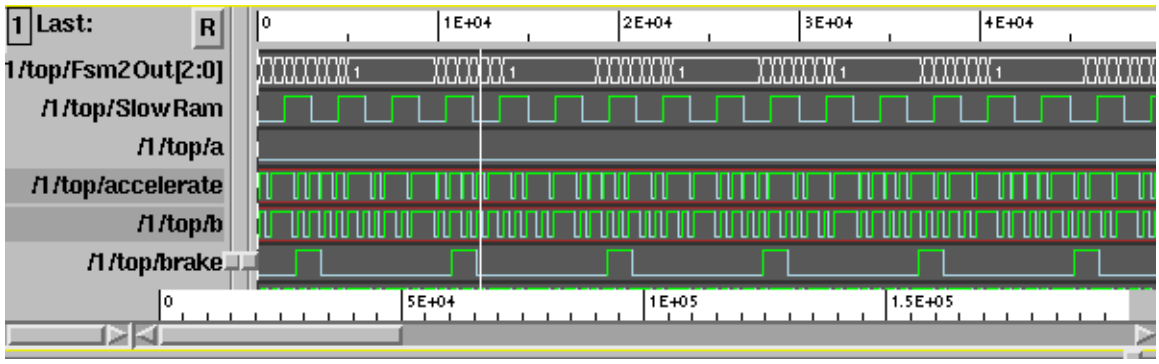


FIGURE 62. Before Fit to Pane

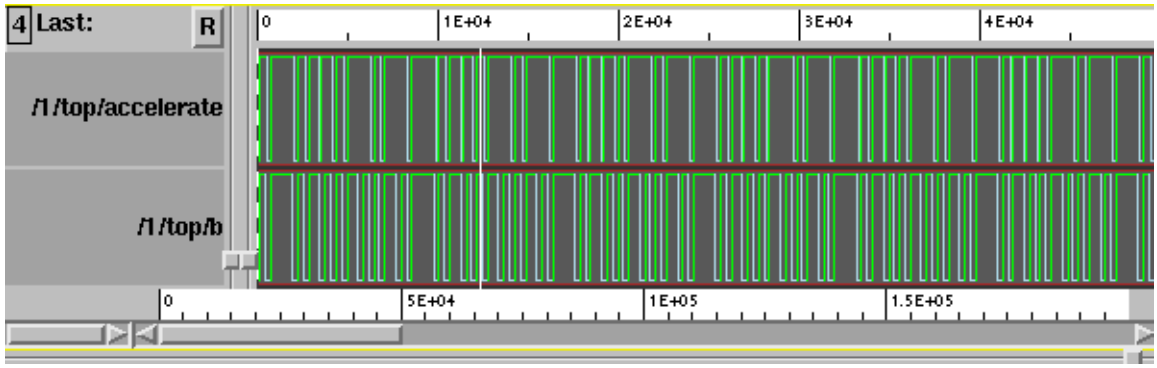


FIGURE 63. After Fit to Pane

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Horizontal Scrolling

TABLE 28. Horizontal Scrolling

Menu Item	Description
Auto Scroll	• When Auto scroll is on, the small toggle indicator on the left side of this button is depressed. • When Auto Scroll is off, this toggle indicator is not visible. This function automatically scrolls the **Waveform** window to the left every two seconds. Use the other buttons in this Horizontal Scrolling function to set how far the window scrolls. You can also use this automatic scroll feature with **Real Time Mode**.
Scroll LE	You can set the horizontal scroll arrows (the triangle buttons on the ends of the horizontal scroll bar) so that each click of the left or right scroll arrows scrolls the **Waveform** window by one leading edge of the signal. 1. Select a signal on the waveform display window. 2. Select Scroll LE. Use this operation to exact the distance in clocks on the **Waveform** display window. For example: 1. Select a clock signal. 2. Set **Scroll LE** for this signal. 3. Set **Snap** in the Icon bar. 4. Snap a cursor to the leading edge of this signal. 5. Select **Fix Cursor**. 6. Click the **right scroll arrow** five times. The waveform window scrolls past the cursor, exactly 5 clocks.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 28. Horizontal Scrolling (Continued)

Menu Item	Description
Scroll TE	You can set the horizontal scroll arrows (the triangle buttons on the ends of the horizontal scroll bar) so that each click of the left or right scroll arrows scrolls the **Waveform** window by one trailing edge of the signal. 1. Select a signal on the waveform display window. 2. Select Scroll TE. Use this operation to exact the distance in clocks on the **Waveform** display window. For example: 1. Select a clock signal. 2. Set **Scroll TE** for this signal. 3. Set **Snap** in the Icon bar. 4. Snap a cursor to the trailing edge of this signal. 5. Select **Fix Cursor**. 6. Click the **right scroll arrow** five times. The waveform window scrolls past the cursor, exactly 5 clocks.
Scroll To Max Time	This option sets the right **horizontal scroll arrow**, so that it scrolls the waveform to the maximum time (all of the way to the right) when you click the scroll arrow. Use this option with auto scroll, to display *marching waveforms*, for real time mode or interactive mode operation.
Scroll Percent....	This option sets the **horizontal scroll arrow** to scroll by the specified percent of the Waveform display, whenever you click the left or right arrow.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Vertical Scrolling

TABLE 29. Vertical Scrolling

Menu Item	Description
Scroll To Signal...	This option opens a window that you can use to scroll to any signal in the signal list. 1. Select a signal in the list. 2. Click **OK**. The Undertow window scrolls to the specified waveform.
Scroll Set Interval...	This option sets the scrolling interval. 1. Select a point on the waveform, at the distance that you want the window to scroll. 2. Select Scroll Set Interval.
Scroll 90%	This option sets the vertical scrolling distance. The window scrolls 90% of the viewable area of the display window, every time you click one of the **vertical arrows**.

Set Display Interval

Use this option to manually set the display interval, or to select one of the predefined intervals.

1. Choose the desired interval.
2. Click **Set**.

- If you are using a VCD file, you can set the display interval. The display interval is the amount of data that loads each time you add a signal to the **Waveform** window. If you set the display interval to a wider area, you can scroll in this wider area without a time penalty. However, loading the signal data is a little slower.
- If you are using a Fast or Optimized file, Undertow always loads all data for the signal, because these formats are so fast.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 30. Setting Display Intervals

Attribute	Definition
Minimum Time/X	
Current Display Time/X	Reads data from a file, for the period indicated by the **horizontal scroll bar** on the bottom time scale.
3X Current Time/X	Reads data from a file, for 3 times the period indicated by the **horizontal scroll bar** on the bottom time scale.
5X Current Time/X	Reads data from a file, for 5 times the period indicated by the **horizontal scroll bar** on the bottom time scale.
All Time/X	Reads data from a file, for all time, regardless of the time indicated by the **horizontal scroll bar** on the bottom time scale.
Start	Manually sets the first time point for reading data
End	Manually sets the last time point for reading data

Auto Update

This option automatically loads data, for the area of signal shown in the waveform display.

If you toggle off ***Window => Auto Update***, the display window does not update with the values in the waveform file. Instead, the horizontal scroll bar shows you where the values are located in the simulation time.

This mode was put in when most Undertow users were using the VCD file. This mode allowed scrolling, without the delay of loading waveform data, because the VCD file was very slow to load and display signals. Users could then use the **Update** icon to update the values on the display window, using the values in the VCD file.

Now almost all users use the *Fast file* or *Optimized file* formats, which load and display the data almost instantly, regardless of the file size. So you should not need to use **Auto Update off** and the **Update** icon.

As file sizes of the highly-compressed *Fast* and *Optimized* files become bigger, you might need to use this mode again in the future. That is, you would turn off **Auto Update**, scroll to the proper point in the simulation, and load *only* the data for the scroll area.

Real Time

In Undertow, you can view the output from Verilog while the simulation is running.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1. Start Undertow in real time mode.

   At the specified time interval, Real Time mode checks the size of the loaded file size.

2. If the file has become bigger, Undertow loads the new data.

For more information about **$vtDump**, see .

1. **Start Verilog**.
2. If the source code does not include the **$vtDump;** and **$dumpvars;** statements, type them at the Verilog prompt.
3. **Start Undertow** with the **-r** and **-t** # options, where # is the real-time mode interval in seconds (the default is 10).

   These options open Undertow in real-time mode, with both auto-scroll and auto-update enabled.

4. **Select the Verilog** output file in fast format (VCD filename with .fast extension) and click **Open**.

Undertow can also read the VCD file in real-time mode (if you did not use the **$vtdump;** statement). To do this, when you open the VCD file, use **Correlated Fast File** to create a Fast File. Use this workaround only if the Verilog is not linked to the Veritools *pli* code. This option generate both a VCD file and a Fast File, so you use more disk space.

You can also use one of the following two methods to create the Fast File while Verilog is running, without starting Undertow:

- Include the **$vtDump;** statement before the **$dumpvars;** statement.

  Verilog creates a Fast File, without creating a VCD file. This process might slow down the Verilog simulation, but it uses much less disk space. If disk space is an issue, you can use the **Optimizing_tool** (PLI routine), which compresses the binary output to one-seventh the size or smaller, compared to the VCD file. For more information, refer to or contact your Veritools representative.

- Include the **$dumpvars;** command, but not the **$vtDump;** command, and start Verilog. Then from your Undertow directory, start the **vdump** conversion utility, with the following arguments:

  vdump -Rcxn [*VCDfilename*]

  The **vdump** utility generates a fast file, using the same name as the VCD file, but with the .fast extension.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To stop **vdump** at the end of the Verilog simulation, type Ctrl-C in the same shell where you started **vdump**.

Note: This second method is faster than the first method, but it generates a VCD file along with the Fast File.

Time Scale

Time Scale opens the Time Scale window. Use this window to select the time units that Undertow uses.

Autoscale uses a three-digit representation of the time value, followed by the appropriate time scale—for example:

- **s** for seconds
- **us** for microseconds
- **ms** for milliseconds

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

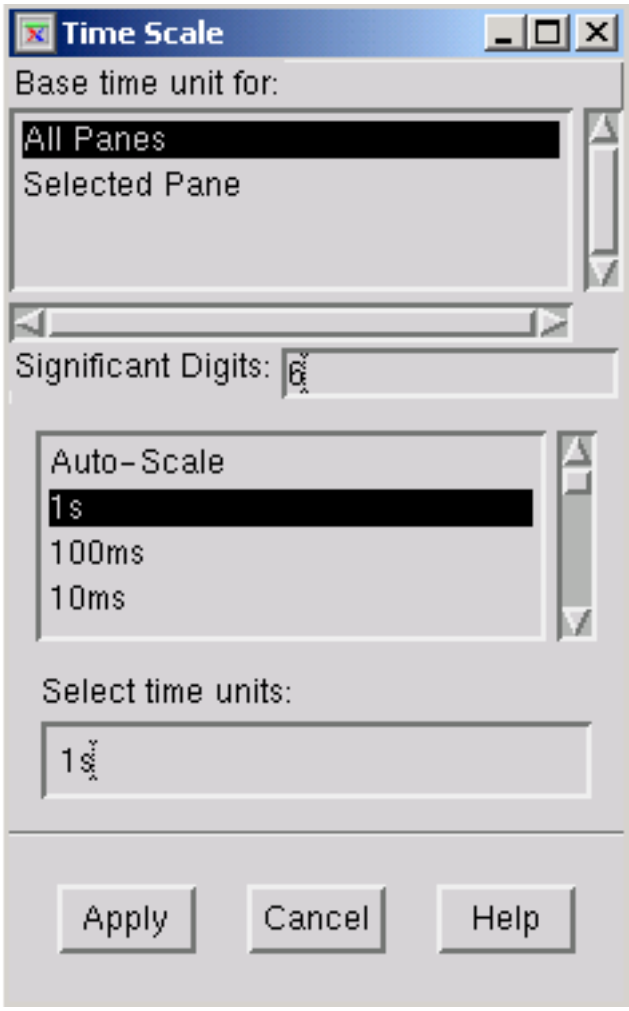


FIGURE 64. **Time Scale Dialog**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Waveform Heights

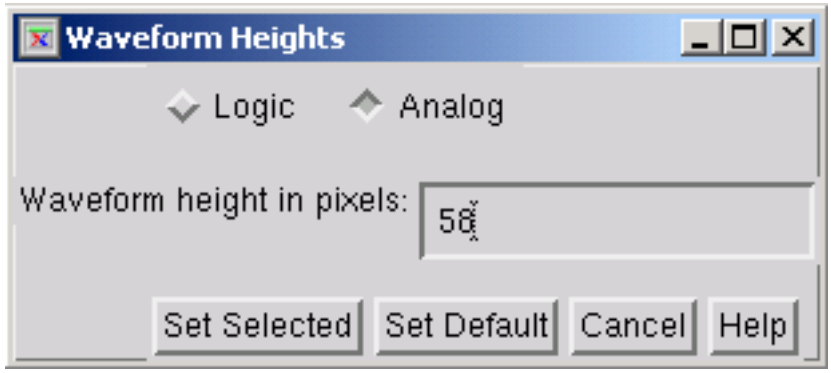


FIGURE 65. **WaveForm Heights Dialog**

Use the **Waveform Height** option to set the height of waveforms, either individually or globally.

- **Set Selected** changes the height of *only* the selected signals.
- **Set Default** affects any signal that you add to the display, but does not change any signals that Undertow already displays.

Sync Panes

Sync Panes synchronizes the horizontal scroll bars between the panes. It has two settings:

- **Sync on Start end point** synchronizes both panes to the same horizontal scroll start and end points, whenever you reposition and release the horizontal scroll bar. All panes that use the same time scales scroll to same point, and have the same zoom.
- **Sync on Center point** synchronizes both panes to the same horizontal scroll center point, whenever you reposition and release the horizontal scroll bar. All panes that use the same time scales scroll to same point, but they can have different zooms.

Add New Pane, Time

Adds a new pane, using the same time axis (x axis) as the current pane.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Add New Pane, Arbitrary X

Adds a new pane, with an undefined x axis. Use this option if the horizontal scale is not a single-precision (32-bit) time value.

Remove Pane

Removes the pane that is in *Focus*,—that is, the pane where the yellow focus button is set.

Adding and Removing Value Area

Adds or removes a new value area in a selected pane.

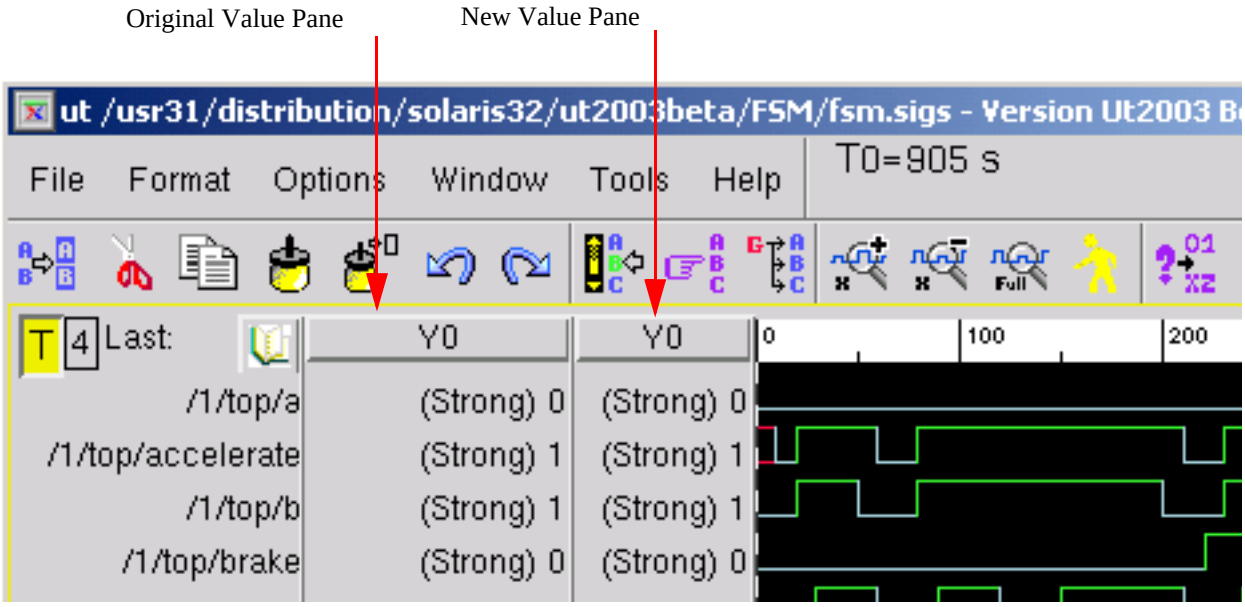


FIGURE 66. **Add New Pane Example**

1. Click the **Value** button to perform various calculations on the displayed waveforms.

   A rectangular menu button appears at the top of the waveform values field.

2. Click the menu button.

   A menu opens, with the following functions:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Y0	Displays the Y value of cursor 0 in the value field.
Y1-Y0	Displays the difference in value between Y1 and Y0.
(Y1-Y0)/(T1-T0)	Displays the slope between cursor 0 and cursor 1.
Average	Displays the average value between cursor 0 and cursor 1.
Slope (T1)	Displays an approximation to the slope at cursor 1.

CURSOR PLACEMENT

Undertow measures between cursor 1 and cursor 0.

1. Click the **Measures** button.
2. Click the **middle** mouse button where you want to place cursor 0.

   Cursor 1 follows the mouse.
3. Click the **middle** mouse button again to place cursor 1.

MOVING CURSORS

To move either cursor 1 or cursor 0 after you place them, click and hold the **middle** mouse button.

Floating Windows

You can tear-off (separate) a menu or icon window. into a window that is independent of the main window.

Options include:

- **Floating Icon Bar**
- **Floating Choose Window**
- **Floating Group Window**
- **Floating Menu Bar**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Tools Menu

Test Analyzer

If you open two Undertow display windows, each with its own file, you can use **Test Analyzer** to see how these files differ.

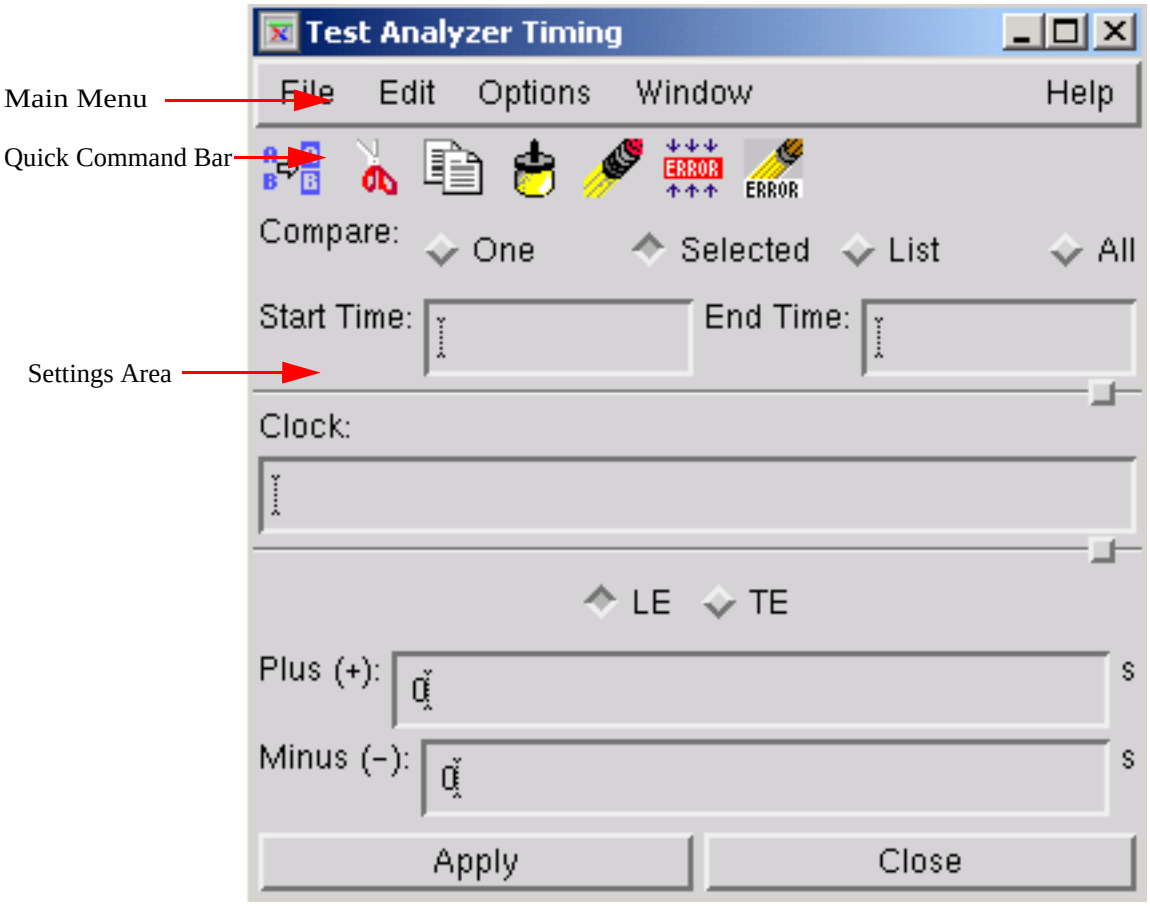


FIGURE 67. **Test Analyzer Dialog**

The main menu in this window has the following sections:

File—saves the current Test Analyzer setup to a file, or loads a previously-saved setup. The Print option prints the contents of the setup file in a readable format.

Edit—lets you select, copy, cut, or paste signals from the display window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Options—has two options:

- **Highlight Errors**—selects the displayed signals that contain errors.
- **Clear Errors**—erases the purple highlighting that marks the errors.

Windows—lets you configure the current window. The configuration dialog box has two input fields:

- **Working Directory—**specifies which directory to save the setup files to or load the files from.
- **File Suffix**—specifies the default suffix that the selection window uses.

You can use two radio buttons to switch the Quick Command Bar, or Icon Bar, to either Glyphs (pictures) or Text.

- The Apply button enables the displayed settings.
- Cancel discards any modifications.

The **Test Analyzer** window has a set of radio buttons that select the signals that the Test Analyzer compares:

One—lets you select one signal from each display window (the names of these signals do not have to be the same), and compare the values of these two signals. To indicate areas of differences between signals, the **Test Analyzer** uses purple highlighting on the signal in the window from which you started the **Test Analyzer**.

Selected—lets you select one or more signals from one display window, and to compare these signals to signals that have the same name in the other display window. The **Test Analyzer Results** window indicates signals that do not match. To highlight the areas of mismatches, select the signals from the list and click **Display**.

List—opens a dialog box, in which you can choose a signal names list. The Undertow ***File => Save [Signal Names]*** menu item creates this list. The **Test Analyzer** compares the signals between the VCD files in the two display windows, and places the names of the signals that mismatch, into the **Test Analyzer Results** window. Use this window to select and display signals that did not match. If the signals are not on the current display window, Undertow adds them, along with the highlighted areas that were incorrectly compared. For example, signals might not be currently displayed in the **Compare list of signals** and **Compare All Signals** operations. If the signals in the list are already displayed, select them from the list, and then click **Display** to highlight the areas that did not match.

All—compares all signals in the two VCD files that you opened in the two different Undertow windows. The **Test Analyzer Results** window shows the names of the signals that did not match. Use this window to select and display signals that did not match. If the signals are not

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

on the current display window, Undertow adds them, along with the highlighted areas that were incorrectly compared. For example, signals might not be currently displayed in the **Compare list of signals** and **Compare All Signals** operations. If the signals in the list are already displayed, select them from the list, and then click **Display** to highlight the areas that did not match

You can use the area below this Compare selection area, to define a timing *window of interest*, which the compare operation uses to define the exact area to compare.

1. Select any signal that has edges that apply to this window of interest.
2. Select the leading or trailing edge of this signal.
3. Define the plus and minus time intervals on either side of this edge, to define the window of interest.

For example:

1. Select a clock.
2. Select the leading edge of the clock.
3. Define the window of interest as plus 5 and minus 25.

   This creates a 30 time unit window, which compares signals 25 time units in front of, and 5 time units after, the leading edge of this clock.

   If the signals differ in this window, the **Test Analyzer** lists them for multiple signals, and highlights the background in *plum*, for the portion of the signal that differs when you display that signal. the **Test Analyzer** ignores differences outside of this window.

*Note: Use **Highlight Errors** (**Error** Icon in the **Test Analyzer** window) and **Go To** Errors (in the **GoTo** window, see Section , "Main Toolbar Icons"), to find errors that might be hard to see in the **Display** window.*

Connectivity Analyzer

Shows connectivity for the selected signals.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 31. Connection Analyzer Menu

Menu Item	Description
Show Connectivity	**History** opens the **Connectivity History** window. Use the window to select a node to return to. The list shows nets that Undertow traversed when it back traced signals. In this window, you can select and return to any net that the **Show Connectivity** window previously displayed. You can also select any net that this window lists, and display its waveform.
Show Hierarchal Names	Lists signals, with their full names and locations through the hierarchy.
Display Drivers, Selected Signal	Lists Drivers for a selected scalar (non-bused) signal.
Schematic Browser	Pops up the **Schematic** Window.

Logic Analyzer

Use this option to search and modify signals by logic states.

Boolean Functions

This option applies boolean expressions to selected signals.

TABLE 32. Logic Analyzer Boolean Functions

Menu Item	Description
New	Creates a new expression.
Modify Selected	Modifies an existing expression.
List	Lists all expressions that you created.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Find Logic

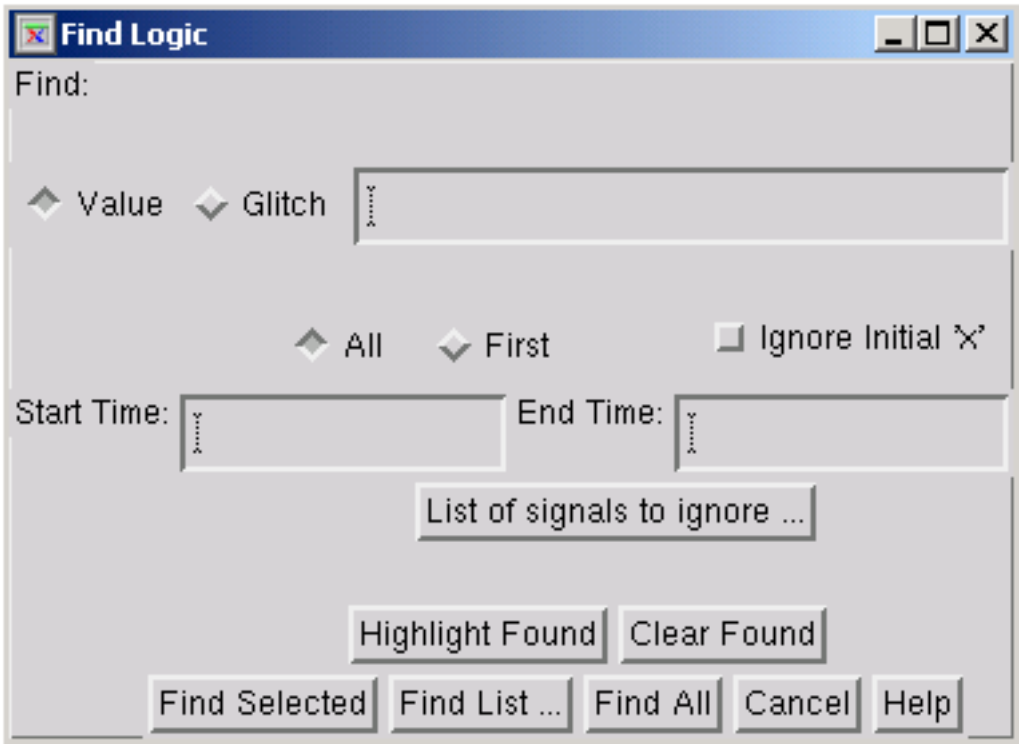


FIGURE 68. **Find Logic Dialog in Logic Analyzer**

Find Logic opens a dialog box that you can use to define parameters for the logic search.

- You can search for signals that have a certain value, and highlight the instances (background in plum color).
- You can also search for glitches that are smaller than the specified time limit.

  Hint: Convert the format to binary, *before* you use **Find Logic**.

A pop-up list shows signals where instances occur. Use this list to choose the signals that you want to display.

- **Find Logic** can highlight all instances, or only the first instance.
- Unlike the **Go To Value** feature, the **Find Logic** option does not accept aliased values. For buses, you must enter a binary value.

The **Find Logic** window has the following buttons:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Value/Glitch—switches between the value and the glitch search mode.

All/First—If **All** is on, then **Find Logic** highlights all instances when it displays the signal. Otherwise, if you select **First**, then **Find Logic** highlights only the first instance of the signal.

Ignore Initial 'x'—ignores the *x* values from the start of the simulation, until the signal goes to either a **1** or a **0**.

List of signals to ignore—opens the **File Selection** window, which lets you select a list of signals to ignore. To generate this file, use the ***File => Save (Signal Names)*** menu option.

Highlight Found—highlights the instances that **Find Logic** found in the waveform display window.

Clear Found—clears the color highlighting from the displayed signals.

Find Selected—runs the test on the selected signals.

Find List—opens the file selection window, which lets you select a list of signals to test. To generate this file, use the ***File => Save (Signal Names)*** menu option.

Find All—runs the test on all signals.

The **Find Logic** window has three input fields:

- Use the top field to define the value to search for, or the minimum hold time (maximum width of a glitch).
- Use the two lower input fields to define a time window for the analysis.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Trigger Conditions

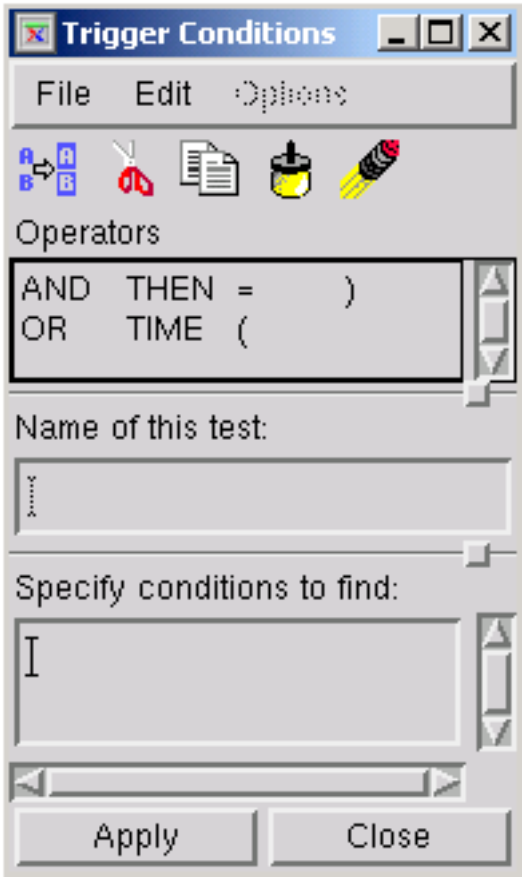


FIGURE 69. **Trigger Conditions New Dialog**

Toggle Analyzer

The Toggle Analyzer option pops up a dialog box, in which you specify the start and end time for the toggle analysis. Table 33, “Toggle Analyzer Dialog Box” describes the options in the **Toggle Analyzer** window.

TABLE 33. **Toggle Analyzer Dialog Box**

Attribute	Description
Start Time	Specifies the start time for analysis.
End Time	Specifies the end time for analysis.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 33. Toggle Analyzer Dialog Box (Continued)

Attribute	Description
List of signals to ignore	Opens the **File Selection** window, in which you can select a list of signals to ignore. To create this list, use the ***File => Save (Signal Names)*** menu option.
Find Selected	Runs the test *only* on the previously-selected signals (except buses).
Find List	Opens the **File Selection** window. Use this window to select a list of signals to toggle. To create this list, use the ***File => Save (Signal Names)*** menu option.
Find All	Runs the test on all the signals in the design (except buses) Use this feature to quickly find signals that did not toggle during a specified lapse of time.

Sequence Analyzer

Use the **Sequence Analyzer** to view a sequence of events at a specific time point, in graphical format. This function requires a *.vcd file.

Analog Analyzer

Interpolate Voltage

For voltage waveforms, lines can be drawn at any angle. This function smooths out analog waveforms that Undertow created from a finite sample.

Interpolate Current

For current waveforms, lines can be drawn at any angle.This function smooths out analog waveforms that Undertow created from a finite sample.

Overlay Waveforms

Opens the **Overlay Analog Signals** window. Use this window to overlay analog waveforms.

1. Paste the signal names to superimpose, into the editing area.
2. Click **Add** to display the signals and overlays.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Calculate Average

This option calculates the average of a waveform. It calculates the integral over a certain time period, and divides the integral by the time period itself. To perform this operation:

1. Place time cursor 0 at the beginning of the area that you want to average.
2. Place time cursor 1 at the end of the area that you want to average.
3. Select **Calculate Average**.

For more information about time cursors, see Section , “Time Cursors”.

Calculate Slope

This option calculates the slope of the selected waveform, at a point that you specify. To perform this operation:

1. Select the waveform whose slope you want to calculate.
2. Place time cursor 0 at the point where you want the slope.
3. Select **Calculate Slope**.

Create Eye Diagram

This option opens the **Eye Diagram Signals** window. Use this window to overlay a single analog waveform onto itself, shifted by the fold-over time.

1. Select the signal name to use as the basis for the eye diagram.

   Undertow automatically lists this signal in the signal area.
2. To choose a color for the diagram, select a color name from the list of colors in the dialog.
3. To display the diagram, click **Apply**.

- **Fold over Time** is the amount by which each overlay (of the same signal) is shifted forward in time.
- **Start Time** is the starting time for the eye diagram.
- **End Time** indicates that there is no further overlaying after this time.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Analog Functions

Opens the **Boolean Functions** window. This window has the following areas:

- **Main Menu** is at the top.
- **Quick Command Bar** is under the main menu.
- **Operators and Functions** lists the Name input field
- **Analog Expression** input area.
- **Actions** buttons:
    - **Apply** creates the new analog waveform, based on the current analog expression.
    - **Close** closes the Analog Functions window.

The main menu in this window has the following sections:

File—saves the current analog expression to a file, or loads a previously-saved expression. The Print option prints the analog expression in a readable format.

Edit—lets you select, copy, cut, or paste signals from the display window.

Window—lets you configure the current window. The configuration dialog box has two input fields:

- **Working Directory**—specifies which directory to save the setup files to, or load the setup files from.
- **File Suffix—**specifies the default suffix that the **File Selection** window uses.

You can use two radio buttons to switch the **Quick Command Bar** to either Glyphs (pictures) or Text.

- The **Apply** button enables the displayed settings.
- **Cancel** discards any modifications.

To construct an analog expression:

1. Paste signals into the Analog expressions area.
2. Combine these signals. You can either:
    - Use operators that you pick from the **Operators** list, or

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- In the **Name** input field, enter a name for the new signal. Then click **Apply** to apply functions to the analog signal.

Analog Functions, Selected—Use this option to modify an analog function that you previously created. For more information about analog functions, see Section , "Application Note - Analog Features".

Application Note - Creating A Fast File

Description

A **Fast File** is a format that Veritools has developed, to increase the performance of Undertow and Undertow Suite. If you use this **Fast File** format, you can increase the performance of Undertow by as much as 100,000 times, compared to conventional **vcd** file formats.

You can use any of three methods to create a **Fast File**:

- From the user interface.
- Use the Veritools **vdump** utility from the command line.
- Call **vdump** from the users PLI.

Creating A Fast File from the User Interface

From the Undertow menus, select **File => Open/Close**.

The **Open Dump File** dialog opens, as shown in Figure 70, "Open Dump File Dialog".

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

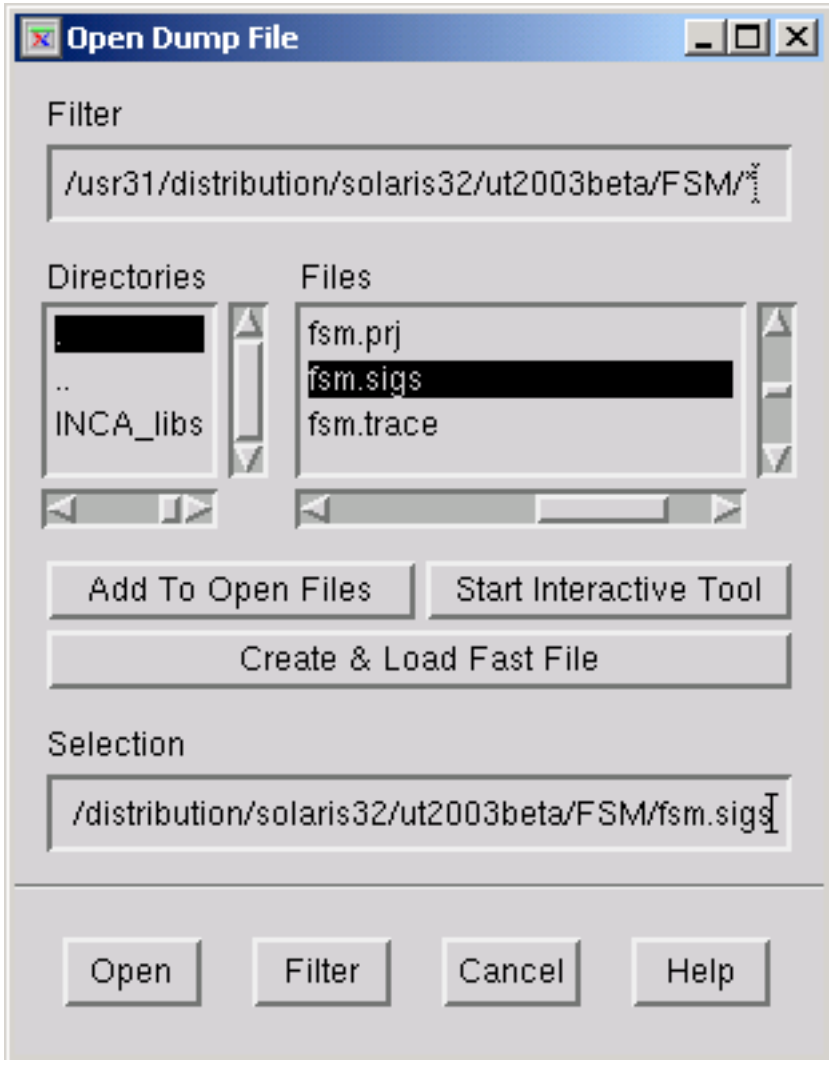


FIGURE 70. **Open Dump File Dialog**

1. Select the **vcd** file of interest.
2. Click on **Create & Load Fast File**.

   Undertow creates a new file, in the same directory, using the same file name but with a .fast extension. In the example in Figure 70, the Fast File would be called *cray.vcd.fast*.

Using The Vdump Routine

To create a Fast File from either a command line or a script, use the **vdump** executable, which is located in your Undertow install directory.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

If you set the **UT_ROOT_DIR** environment variable, as described, then add this variable to your PATH—for example:

export PATH=$PATH:$UT_ROOT_DIR

If the file is in a format other than Verilog VCD, use either the Create Fast File or the Create & Load Fast File button to convert the file to the Fast File format. Currently, you can convert files from EPIC Powermill/Timemill, Hspice *wsf*, or Mentor ASCII format, into the Veritools Fast File format.

You can also use the following command to create a Fast File:

vdump -xcu [*dumpfile_name*]

The command line options for using **vdump** are:

Usage: **vdump [-vRxcun] [[-[r|b|m|t|T] #]...] [-o *outfile*] *dumpfile***

Dumpfile Boolean key definitions:

-v:	Enables verbose output to standard error.
-R:	Runs in real time mode.
-x:	Does not exit on errors, if possible.
-u:	Unlinks (removes) the original file when done.
-n:	Assigns a Fast File name, based on the input file name plus the .fast file extension.

Argument descriptions and defaults:

-b:	Primary block size for information about signals.
-m:	Secondary (more) block size for information about signals.
-t:	Specifies the start time for reduced file formats.
-T:	Specifies the end time for reduced file formats.
-r #minutes:	(default=10 minutes).
-F #seconds:	Specifies how much time to wait before flushing the outfile (default=disabled).
-o outfile:	(default=ut.dump).
dumpfile:	VCD file name.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Generating A Dump File (VCD) From Verilog

The Verilog **$dumpvars** command generates the **Value Change Dump File** (VCD). This file contains two main parts:

- The definitions section defines the hierarchy of the design, and all signals defined in the design.
- The data section defines the changes in those signals over time.

The **$dumpfile** system task specifies the VCD file name. This task uses only one parameter, a string containing the name of the file to create. If you omit this task, the name defaults to **verilog.dump**. Execute this command once, before any **$dumpvars** tasks.

To specify which variables to dump, the **$dumpvars** task can include zero, one, or two parameters.

- If you do not include any parameters, then Undertow dumps all variables in your design.
- To dump all variables in a specific module and all its submodules, use the module instantiation name as a parameter in the **$dumpvars** command.
- To dump all variables in a module, down to a specific level of submodules, execute the $dumpvars command with two parameters:
  - The number of levels to dump.
  - The module to dump.

You can execute as many **$dumpvars** commands as you need, but you must execute all of them at the same simulation time.

For example:

```
initial begin
        // dump values to mydesign.dump
        $dumpfile (mydesign.dump);
        // dump all variables in module adder_2
        // and all its submodules
        $dumpvars(adder_2);
        // add all variables in module adder_3 only
        $dumpvars(1, adder_3);
        // add all variables in module adder_4 and
        // only the submodules adder_4 instantiates
        $dumpvars(2, adder_4);
        // add all of the variables in the design
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
        $dumpvars;
end
```

Veritools provides a utility that you can use to specify the maximum size of the VCD file. You can include the **$utLimitVCD(#)** command in the Verilog source, to limit the size of the VCD output file to the specified number (#) of Megabytes.

Generating A Fast Dump File From Verilog

To generate a Fast File directly from Verilog, adding the **$vtDump** statement before the **$dumpvars** statements.

*Note: Do not forget to remove the **$utLimitVCD(#)** statement also!*

Verilog then invokes the vdump file conversion utility, which receives the VCD output through a named pipe, and generates a Fast File on the fly. For the example in the preceding section, the simulation creates two output files:

- **mydesign.dump** (pipe type, 0 bytes size; this file is removed when you exit the simulator)
- **mydesign.dump.fast**, which Undertow can read directly.

You can view the output in Undertow while it is being generated. (refer to Section , “Real Time”).

For an example of how to use the Veritools PLI routines, refer to the **README_PLI** file and the **qtest.v** Verilog source file sample, included in the **pli_example_files** directory.

Another way that you can create output from Verilog is to use the **Optimizing_tool** (PLI routine) to generate a compressed file. This file also contains the signal strength information.

1. Make sure that you have linked Veritools PLI to the Verilog.
2. Use the **$vtDumpvars** command.

   The **$vtDumpvars** system task was designed to behave exactly like **$dumpvars**. Refer to .

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Application Note - Analog Features

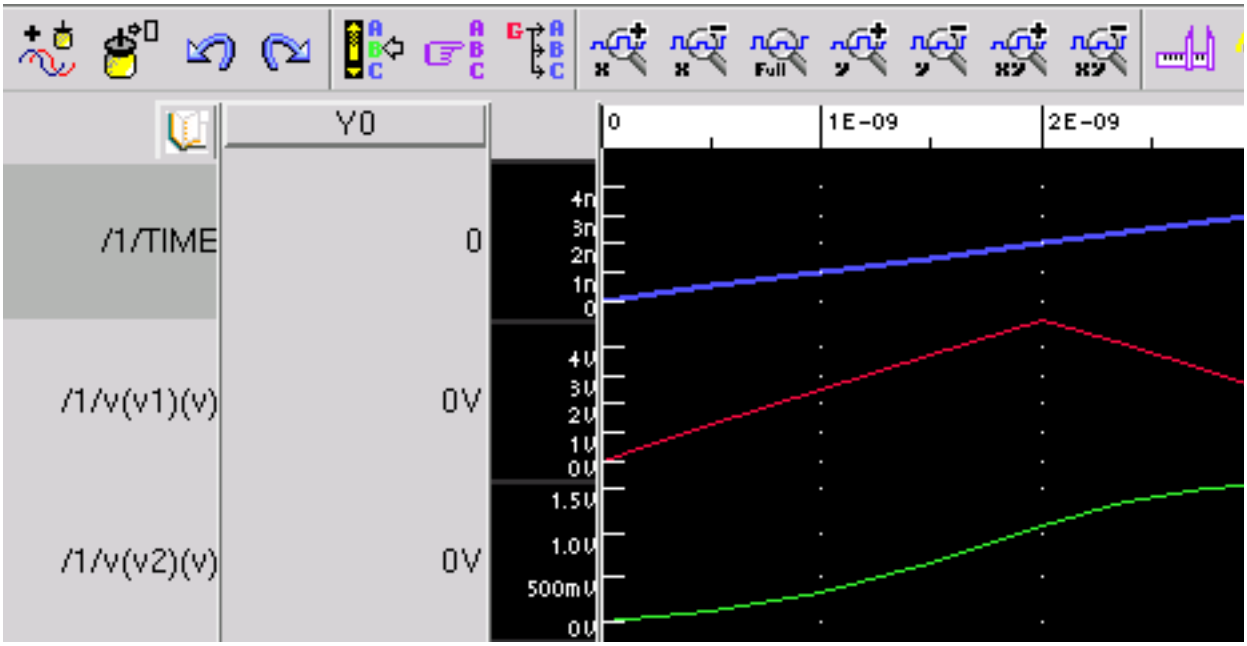


FIGURE 71. Waveform View With Analog Signals

Description

Undertow includes a powerful analog analysis setting that displays and analyzes the results from analog simulations. Undertow automatically enables these analog features when you load an analog Waveform (for instructions on how to load Waveform files, refer to Section , “Wave Display”).

Figure 71, “Waveform View With Analog Signals” shows Undertow displaying an analog file. It also shows the analog-specific buttons in the main toolbar.

- For a description of these analog buttons, see Section , “Main Toolbar Analog Icons”.
- The regular toolbar buttons are described in .

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Main Toolbar Analog Icons

This section describes the analog function buttons in Undertow. For a description of the regular Toolbar buttons, see .

Note: *These icons are visible only when you load analog files in the* ***Waveform*** *viewer.*

TABLE 34. Analog Buttons

Button	Description
	Paste, New Overlay—Inserts a signal into the **Waveform** viewer, above the currently-selected signal. This option adds the last signal in the cut or copy buffer.
	Paste, Add To Overlay—This option overlays the signal in the copy or cut buffer on top of the currently-selected signal.
	Zoom In, Vertical—zooms in, in the vertical direction only.
	Zoom Out, Vertical—zooms out, in the vertical direction only.
	Zoom In, Both—zooms in, in both the vertical and horizontal directions.
	Zoom Out, Both—zooms out, in both the vertical and horizontal directions.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 34. **Analog Buttons (Continued)**

Button	Description
	Measures—If you toggle this button on: • When you click the left mouse button inside the signal viewer, an **X0** marker appears. • When you move the mouse anywhere to the right, and then click again, an **X1** marker appears. This **X1** marker follows the mouse cursor. • A third left mouse click freezes the **X1** marker at the position clicked. To move the **X1** marker, click the left mouse button again. These markers, and the deltas between them (x and y axis delta), display above the signal viewing area, just to the right of the **Tools** menu button.
	Trigger -If you click this button, Trigger menu will appear and user can specify an exact value to anchor at the waveform window. Users can also specify delta x or delta y. Pressing **Next** can trigger another set of data
	Display, Add Plot—This option prompts you for the signal to use as the x axis, then prompts you for the signal to use as the value against the x axis signal. The **Waveform** viewer plots these signals.It can also Plot FFT/DFT.
	Set Master Signal—When you select this option, any signal that you select in any overlay displays in bright gold, so you can more easily see the selected signal. *Note: You can define only one master signal for each overlay.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 34. Analog Buttons (Continued)

Button	Description
	Fit To Pane—Expands the selected waveform(s) to fit the vertical display distance. Use the vertical scroll buttons to de-select this feature. *Hint:* One easy way to view your analog waveforms is to do the following. 1. Select ***Window =>Waveform Heights***. 2. Set the waveform height to 10 pixels. Your waveforms compress vertically. 3. Select the waveforms that you want to view better. 4. Click **Fit To Pane**.
	Title Tag (Create Tag)—When you click this icon, you are asked to use your left button to place a tag. 1. Left click on the place on the waveform where you want the tag. A box labeled **TAG** appears. 2. To move the tag box, use the left mouse button to drag and drop. 3. To change the text or remove the tag, right-click on the tag.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 34. Analog Buttons (Continued)

Button	Description
	Values Tag (Create Tag, One Point)—When you click this icon, you are asked to use your left button to place a tag. 1. Left click on the place on the waveform where you want the tag. A box with the coordinates for the point appears as a tag. 2. To move the tag box, use the left mouse button to drag and drop. 3. To move the tag anchor point, press the left button on the anchor point, and drag-and-drop it to the new location. 4. To change the text or remove the tag, right-click on the tag. If the Value Tag is time based: • Coordinates for anchor points are shown as **T0** values. • Time is the **X** coordinate. • Value is the **Y** coordinate.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 34. Analog Buttons (Continued)

Button	Description
	Measure Tag (Create Tag, Two Points)—When you click this icon, you are asked to use your left button to place a start tag. 1. Left click on the waveform, where you want the start tag. You are asked to use your left button to place an end tag. 2. Left click on the waveform, where you want the end tag. 3. To move the tag, use the left mouse button to drag and drop. 4. To move the tag anchor point, press the left button on the anchor point, and drag-and-drop it to the new location. If the Measure Tag is time based: • Coordinates for anchor points are shown as **T0** values. • Time is the **X** coordinate. • Value is the **Y** coordinate. If you right-click on the tag, you have the following options: • X1-X0 • Y1-Y0 • (X0-Y0) / (X1-Y1) • (X1-X0) / (Y1-Y0) • (Y1-Y0) / (X1-X0) • Average Y • Set Text • Remove Tag
	Analog Configuration—This option pops up a dialog that you can use to configure analog features. See Section , “Analog Configure Dialog”.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Chooser Tool Bar Icons

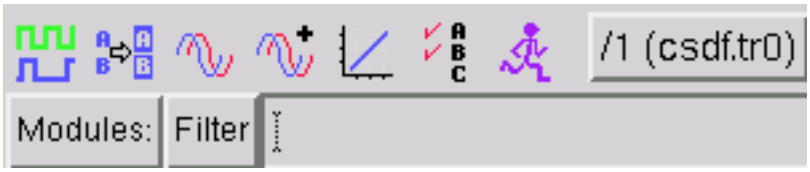


FIGURE 72. **Chooser Tool Bar**

When you toggle on the Chooser icon (in the main tool bar), the Chooser area appears at the bottom of the Undertow **Waveform** window. Table 35, “Chooser Toolbar Analog Icons” describes the icons that the **Analog** mode adds to the **Chooser Tool Bar**.

Note: *These icons are visible only if you loaded analog files.*

TABLE 35. **Chooser Toolbar Analog Icons**

Icon	Description
	Display, New Overlay—If you toggle this option on, any signals that you select in the **Signal Browser** window, display in a new overlay.
	Display, Add Overlay—If you toggle this option on, any signals that you select in the **Signal Browser** window, display either in an overlay, or in the signal currently selected in the **Waveform** window.
	Display, Add Plot—This option prompts you for the signal to use as the x axis, then prompts you for the signal to use as the value against the x axis signal. The **Waveform** viewer plots these signals.It can also Plot FFT/DFT.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

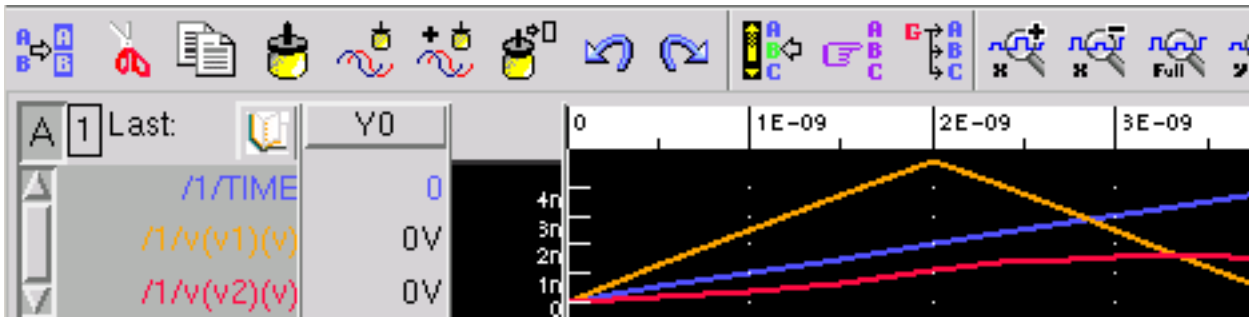


FIGURE 73. **Signals Overlaid**

Analog Configure Dialog

When you click the **Analog Configuration** button (shown at the left), the **Analog Configuration** Window opens (see Figure 74, “Analog Configure General Dialog”).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

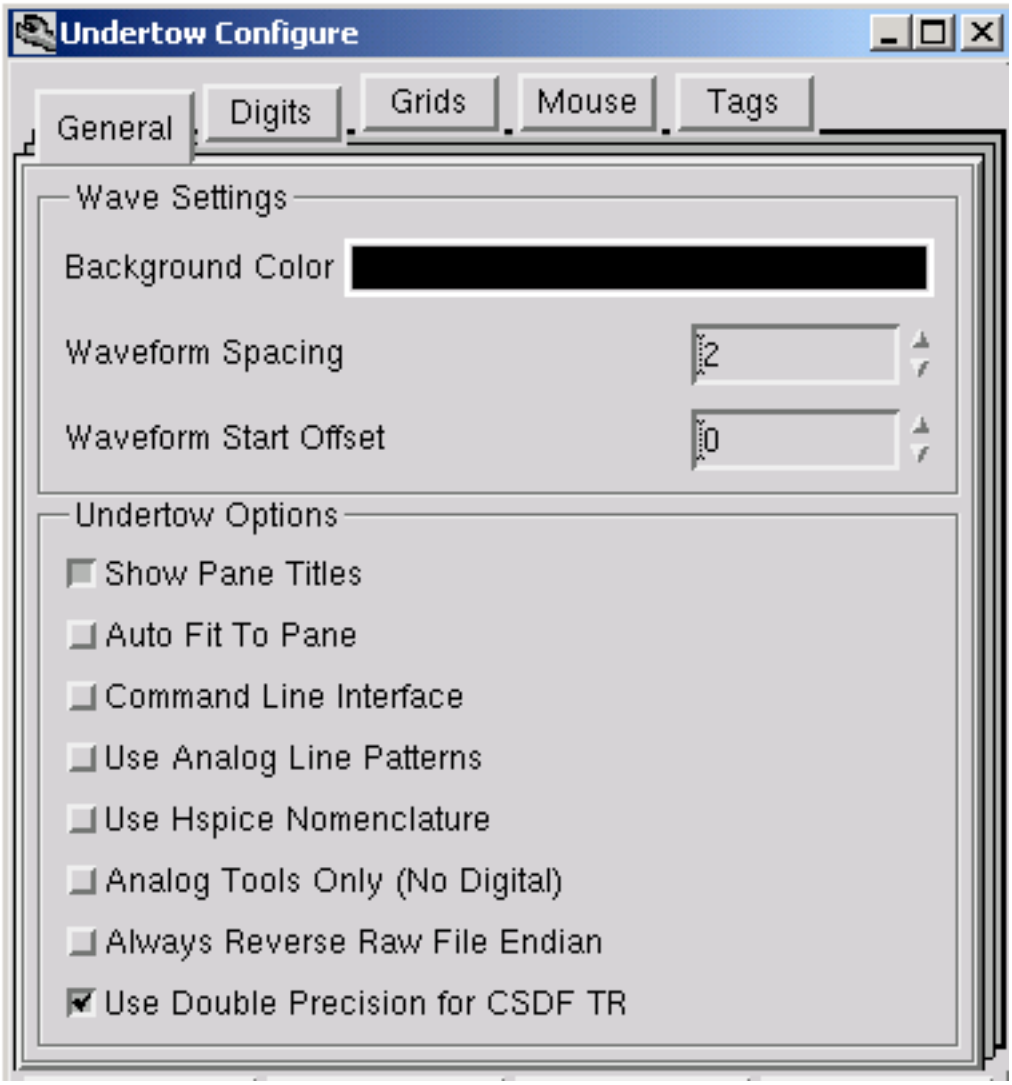


FIGURE 74. **Analog Configure General Dialog**

TABLE 36. **Analog Configure General Dialog Fields**

Attribute	Description
Waveform Background Color	Pops up a list of colors that you can select for the waveform display area. To change the waveform background color, select a color from the list, and click **Apply**. Standard colors are *black* and *grey35*.
Add Color To List	Use this option to add a color to the **Waveform Background Color** popup list. You can add any valid color entry found in the **rgb.txt** file, or any valid hex value (such as *#7799aa*).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 36. Analog Configure General Dialog Fields (Continued)

Attribute	Description
Waveform Spacing	Sets the number of pixels that separate signals, or between waveform overlay panels. The default is 2 pixels. For example, 20 pixels provides about 1/4 inch between waveform overlay panels.
Waveform Starting Offset	Specifies the displacement (in pixels) along the x axis, for the starting point of signals. Use this option to set blank space in front of the displayed waveform. An offset of 20 provides about 1/2 blank space at full zoom; this space increases with each zoom-in operation.
Show Pane Titles	If you select this toggle, Undertow displays a title above each pane, if you created a title for that pane.
Right Click Pops Up Wave Window Menu	• If you toggle this option **on**, you can click the right mouse button in the **Waveform** viewer to pop up a dialog box for zooming and other operations. • If you toggle this option **off**, when you click the right mouse button, a zoom-in operation starts.
Auto Fit To Pane	If you toggle this button **on,** Undertow scales all signals to fit completely in the **Waveform** viewer. Selected signal(s) fill the entire pane.
Command Line Interface	If you toggled this option **on**, a one-line text window appears at the bottom of the Undertow window. You can use this text window to type commands, instead of using the mouse and dialog boxes.
Use Analog Line Patterns	Displays and prints 8 different line patterns, for selected analog signals in the same overlay.
Use HSpice Nomenclature	*Use HSpice Nomenclature* has no immediate effect. It takes effect the next time that you open a file or execute **Reread Dumpfile**. This option applies only to `.ac0` files, and only if there are hierarchical characters (“.”) in the signal names. These become modules if they do *not* use Hspice nomenclature.
Analog Tools Only (No Digital)	Moves all non-analog tools from the Tools menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

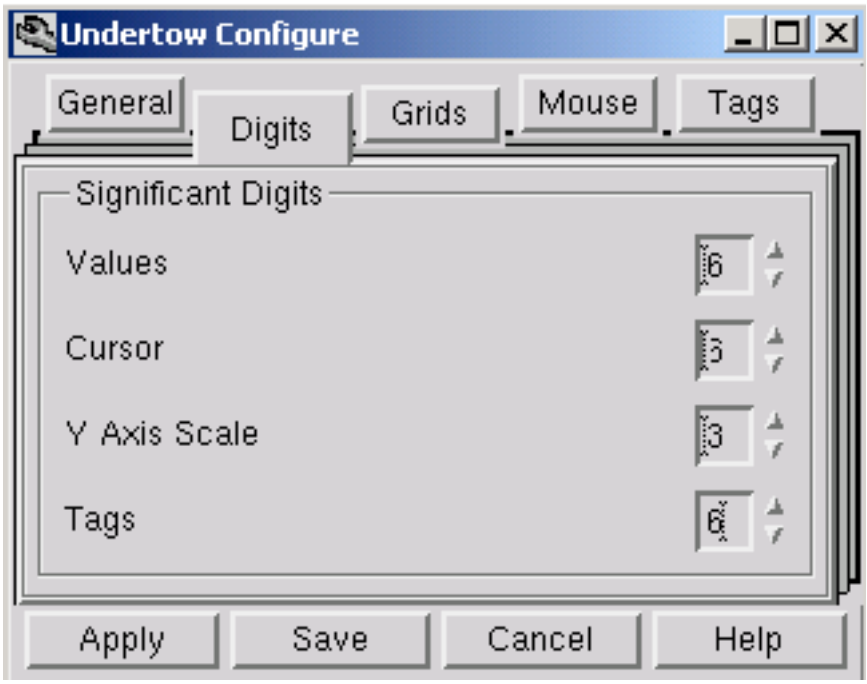


FIGURE 75. Analog Configure Digits Dialog

TABLE 37. Analog Configure Digits Dialog Fields Scales

Attribute	Description
Values Significant Digits	Sets the number of digits that appear in the **Value** area of the **Waveform** display. The **Value** area is located between the signal names and the waveforms.
Cursors Significant Digits	Sets the number of digits that the cursors use. These digits appear at top of the Waveform window, or on cross hairs if you selected the *cross hair value* display.
Y-Axis Scale Significant Digits	Sets the number of digits shown on the Y (Vertical) Scales.
Tag Significant Digits	Specifies the number of significant digits to show for the Tag Values.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 76. Analog Configure Grids Dialog

TABLE 38. Analog Configure Grids Dialog Fields

Attribute	Description
Analog Grids	• If you toggle this option *off*, the **Waveform** viewer uses the default grid spacing. • If you toggle this option *on*, the grid spacing is set as you specified in **Minimum pixels per horizontal grid** and **Minimum pixels per vertical grid**. You can also display the left and right scales.
Minimum pixels per horizontal grid	This option specifies minimum pixels per *horizontal* grid.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 38. Analog Configure Grids Dialog Fields (Continued)

Attribute	Description
Minimum pixels per vertical grid	This option specifies minimum pixels per *vertical* grid.
Show left scale	Flags for showing the left scale.
Show right scale	Flags for showing the right scale.
Minimum pixels per major scale division	This option specifies the minimum pixels per *major* scale division.
Minimum pixels per minor scale division	This option specifies the minimum pixels per *minor* scale division.

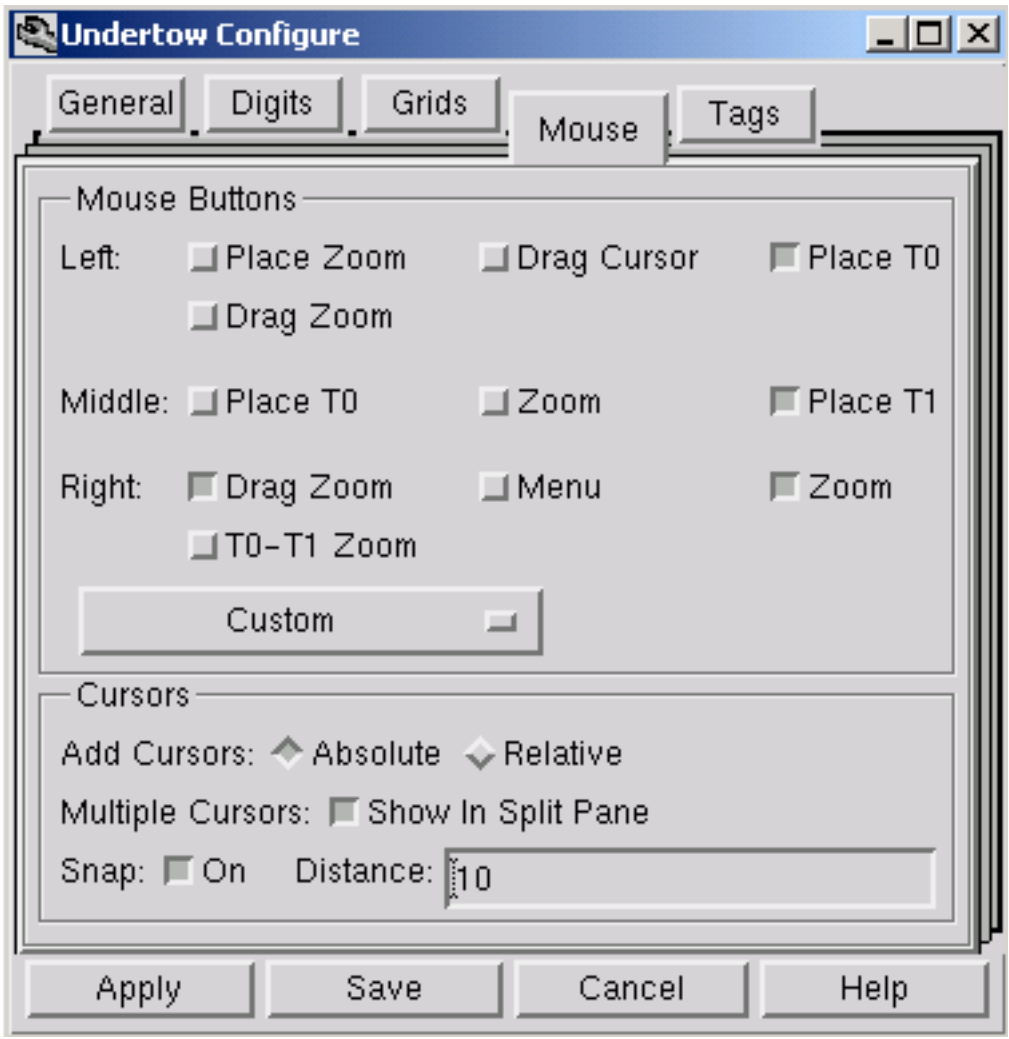


FIGURE 77. Analog Configure Mouse Dialog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 39. Analog Configure Mouse Fields

Attribute	Description
Left Mouse	Can be set to provide the following operations; *Place Zoom*, allows user to place a dashed line for the left and then the right zoom boundaries, *Drag Cursor*, after pressing Add Cursor, this allows the user to drag and place this cursor *Place T0*, allows the user to place the T0 cursor when pressed
Middle Mouse	Can be set to provide the following operations; *Place T0*, places the T0 cursor when the middle button is pressed *Zoom*, does a zoom operation when the middle mouse button is pressed *Place T1*, places the T1 cursor when the middle button is pressed
Right Mouse	Can be set to provide the following operations; *Drag Zoom*, press down this mouse button, slide and zoom over slide area when released *Menu*, if pressed down and held a menu comes up with window operations that can be preformed *Zoom*, does a zoom when right mouse button is pressed *T0-T1 Zoom*, zooms between T0 and T1 when right mouse button is pressed

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

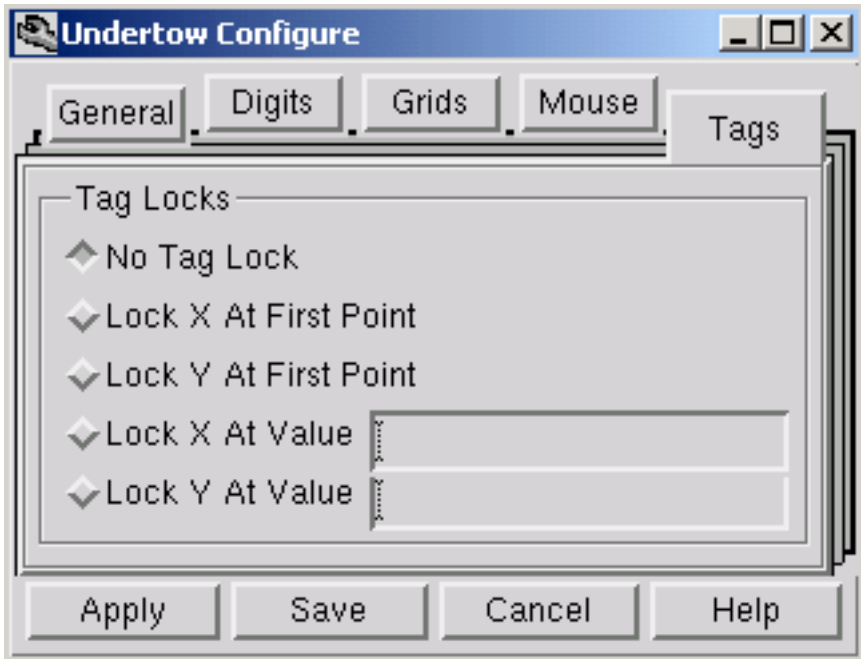


FIGURE 78. Analog Configure Tags Dialog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 40. Analog Configure General Tags Fields (Sheet 1 of 4)

Attribute	Description
No Tag Lock	Use this option to place tags on any point on any signal. • If you select a signal in the selected overlay, the cursor target appears only on the selected signal or signals. • If you do not select a signal in the selected overlay, the cursor target appears on all signals in this overlay. *Note: This cursor target action is the same for this mode, and for all modes listed in this table.* 1. Click on the **Measures Tag** or **Value Tag** icon. 2. When the cursor target is visible, click the left mouse button. The first anchor target is set. 3. Click the left mouse button to set the second anchor, if required. The tag automatically appears between the anchor cursors. 4. Repeat this process as often as you need to. 5. To end the process, click the **Close** button on the **Tag** popup dialog box.
Lock X At First X Point	You can place tag anchor points for the **X** axis. The first anchor point that you set, sets all other anchor points to this same value. *Note: The anchor target displays only when the cursor arrow is over the value for the **X** first point for any signal, indicating that you can place the tag anchor at this point.* If you set this mode while the **Measure** or **Value** tag is toggled on, then the cursor tag is visible *only* until you click the left mouse button, which places the first tag anchor point. After that, the cursor target is visible only at the same **Y** value on different signals in this overlay. If a cursor target appears over the same **X** value on another signal, you can use the left mouse button to place and set the second anchor point. The tag appears between the two anchor points.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 40. Analog Configure General Tags Fields (Sheet 2 of 4)

Attribute	Description
Lock Y At First Y Point	You can place tag anchor points for the **Y** axis. The first anchor point that you set, sets all other anchor points to this same value. *Note: The anchor target displays only when the cursor arrow is over the value for the **Y** first point for any signal, indicating that you can place the tag anchor at this point.* If you set this mode while the **Measure** or **Value** tag is toggled on, then the cursor tag is visible *only* until you click the left mouse button. After that, the cursor target is visible only at the same **Y** value on different signals in this overlay. If a cursor target appears over the same **Y** value on another signal, you can use the left mouse button to place and set the second anchor point. The tag appears between the two anchor points.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 40. Analog Configure General Tags Fields (Sheet 3 of 4)

Attribute	Description
Lock X At Value	When you place the tag anchor points, this option sets all anchor points to *only* this **X** value. *Note: The anchor target displays only when the cursor arrow is over the value for the **X** first point for any signal, indicating that you can place the tag anchor at this point* 1. If you sets an **X** value in the text area, set this mode, and toggle on the **Measure** or **Value** tag, then the cursor tag is visible whenever the cursor target is over this value on a signal. 2. If the cursor target is visible, click the left mouse button to place the first tag anchor point. The cursor target is visible *only* on the same **X** values on any signal. 3. You can set the second anchor point *only* when the cursor target is visible on the signal. The tag appears between the two anchor points. *Note: Set the **X** value first, and then set this mode. If you set the mode **before** you set the value, then the value does not take effect until you do one of the following:* • Click Apply, or • Click OK, or • Toggle this mode *on* again.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 40. Analog Configure General Tags Fields (Sheet 4 of 4)

Attribute	Description
Lock Y At Value	When you place the tag anchor points, this option sets all anchor points to *only* this **Y** value. *Note: The anchor target displays only when the cursor arrow is over the value for the **Y** first point for any signal, indicating that you can place the tag anchor at this point* 1. If you sets a **Y** value in the text area, set this mode, and toggle on the **Measure** or **Value** tag, then the cursor tag is visible whenever the cursor target is over this value on a signal. 2. If the cursor target is visible, click the left mouse button to place the first tag anchor point. The cursor target is visible *only* on the same **Y** values on any signal. 3. You can set the second anchor point *only* when the cursor target is visible on the signal. The tag appears between the two anchor points. *Note: Set the **Y** value first, and then set this mode. If you set the mode **before** you set the value, then the value does not take effect until you do one of the following:* • Click Apply, or • Click OK, or • Toggle this mode *on* again. You must set the units for the **Y** value—for example, **1.75V**, or **2.330ua**. To determine the required units for the text value, press the middle mouse button on the desired point. *Note: If you do not set the units, this feature cannot find the value that you are looking for.*

Cursors

Undertow provides three time cursors. The cursor is not simply where the mouse is pointing. You can use these cursors for temporary marks and measurements.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- To move cursor **T0**, press the middle mouse button.
- To move cursor **T1**, press both the Control key and the middle mouse button at the same time.
- To move cursor **T2**, press both the Shift key and the middle mouse button at the same time.

*Note: You can access the **T1** and **T2** time cursors only if both **NUM LOCK** and **CAPS LOCK** are turned OFF.*

The top of the screen shows values for these cursors, and measures differences between them. The ***Window* => *Cursors*** pulldown menu provides some useful features.

Examples

1. Click on the y-axis to select that signal.
2. Place the mouse cursor in the center of the waveforms, and hold down the middle mouse button.

   The **T0** cursor appears.

   - To place the **T1** cursor, use the middle mouse button while holding down the **<Shift>** key.
   - To place the **T2** cursor, use the middle mouse button while holding down the **<Ctrl>** key.
3. Move the mouse around.

   The time value changes at the top of the Undertow window, and the signal value changes on the left.
4. Release the middle mouse button, to drop the cursor at a particular time.

Zooming

To perform any zoom, click on the appropriate signal(s) first, and then click the appropriate zoom button.

To zoom-in the x axis:

1. Click the left mouse button, to place the first zoom cursor.
2. Click the left mouse button again, to place the second zoom cursor.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Click the right mouse button, to zoom the Undertow window to the boundaries of the zoom cursors.

Window Sizing

1. Use the ***Window =>Waveform Height*** pulldown menu to set the default analog and digital waveform height, and to set the height for any current waveform.

2. Use the **Fit To Pane** icon (on the Undertow toolbar) to size the current waveform pane to full.

Another way to change the waveform height, is to add the following line in your .Xdefaults file:

UNdertow.WaveHeight: *X*

where *X* is the number of pixels to use as the waveform height.

Pane Zoom Analog

This options sets the pane defaults for the pane that is in focus. Use this window to set both the bottom (minimum) values for each overlay area, and the top (maximum value) for each overlay in the focused pane, for the following scales: Use the ***Option***s=>***Pane Control Pane***=>***Zoom Time*** to get to this window.

- Time/X
- Voltage
- Current
- Power

You can also set the units for each of these parameters.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

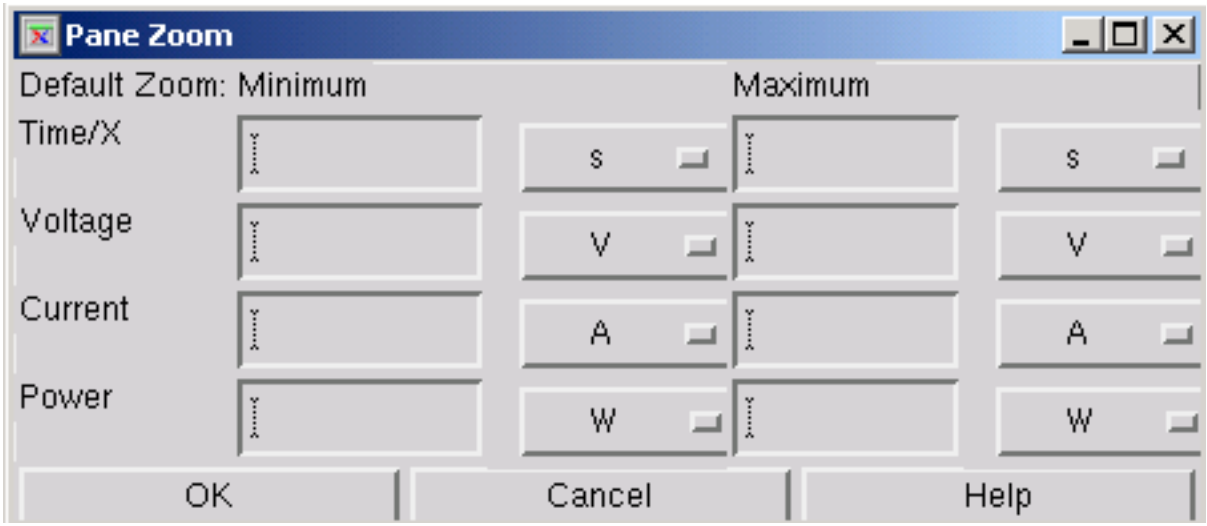


FIGURE 79. Pane Time Zoom

The Waveform display initially defaults to **Normalize** zoom. This feature determines the *min,max* of the signal, and sets the zoom so that the signal fills all of the overlay zoom area, from top to bottom. You can then set this window as the default.

For individual overlays, you can use **Selected Analog Waveform Properties** to override any prior default zoom. You can then set the final zoom for any overlay or overlays.

Measures

You can use cursors to perform quick measurements.

1. To start the basic measure command, select the **Measures** icon from the Undertow button bar.
2. Use the middle mouse button to select two points on the waveform.

   The top of the window shows numerical values, for each of the two points that you selected, and for the difference between them.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

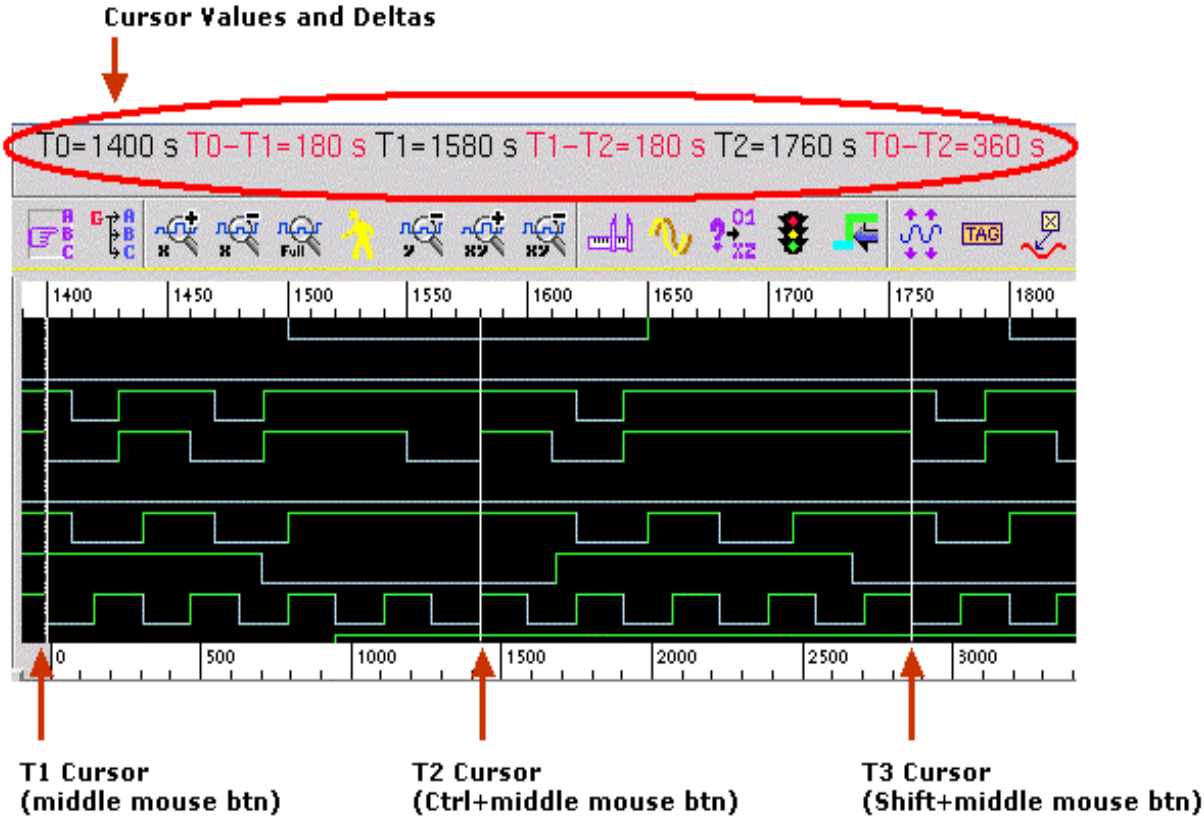


FIGURE 80. Measure example

Calculate Slope

1. To set the first point, **T0**, press the middle mouse button.
2. To set the second point, **T1**, press both the middle mouse button and the <Shift> key.

   The top of the window shows numerical values (0 to 459100), for each of the two points that you selected.
3. Select ***Tools*** => ***Analog Analyzer*** => ***Calculate Slope*** from the menus.

   The **Output** window pops up.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

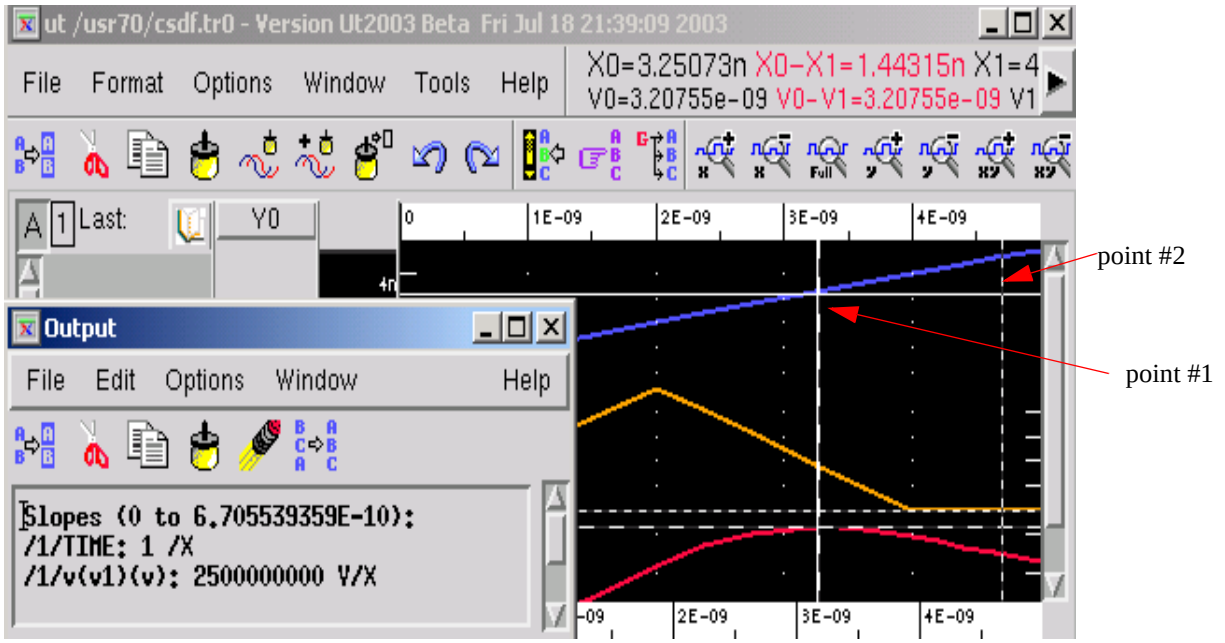


FIGURE 81. Calculate Slope Example

Overlay Waveforms

You can use this option to overlay analog waveform display, within the same panel of the Undertow **Waveform** viewer.

1. Select ***Tools*** => ***Analog Analyzer*** => ***Overlay Waveforms*** from the menus.

   The **Overlay Analog Signals** window pops up.

2. Press the middle mouse button, to drag and drop the signal names that you want to superimpose in the editing area.

3. Click **Add** to display the signals.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

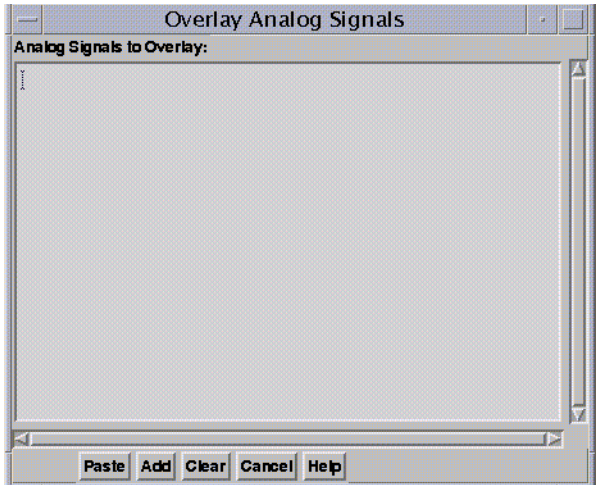


FIGURE 82. Overlay Analog Signal

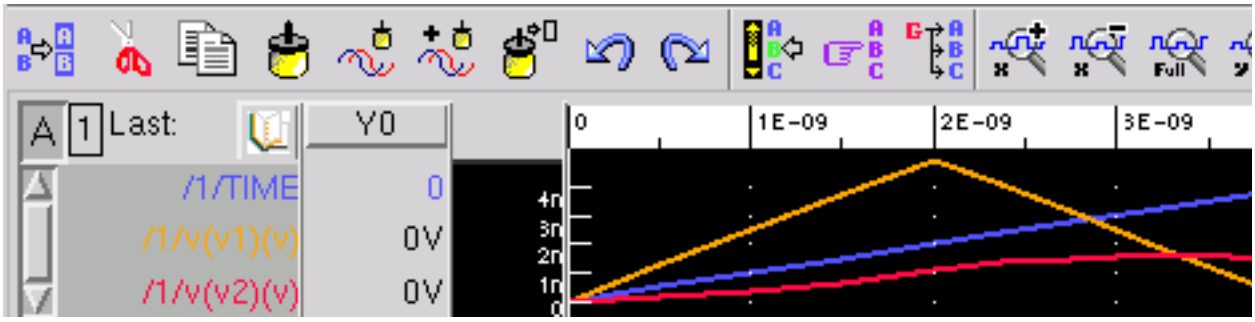
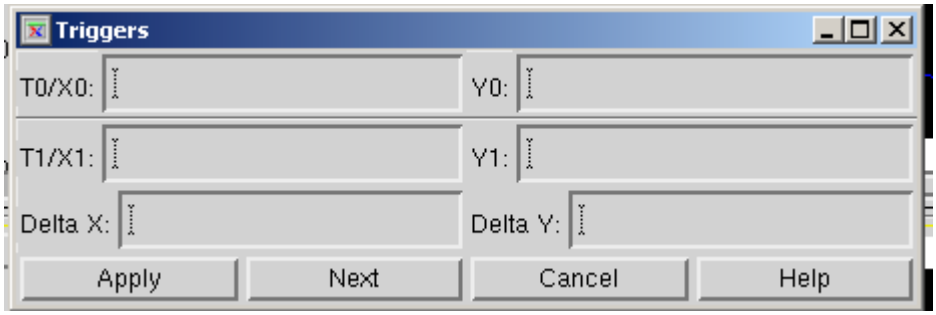


FIGURE 83. Signals Overlaid

4. *(Optional)*. To remove a signal from the overlaid signals, click the middle mouse button on the signal.

Trigger Functions

Trigger functions allow user to slap to a specific point



 Copyright Veritools 2004 All Rights Reserved

Triggers Menu

T0/X0, T1/X1 - value that wish to locate in X axis

Y0, Y1 - value that wish locate at Y axis

Delta X - Delta X that wants to locate from T0,

Delta Y - Delta Y that wants to locate from Y0.

Users needs to set either of these 2 values to allow waveform to trigger at the correct position.

For example: specify Y0 = 0 , Y1 = 0, UT will find the 2 points which Y=0.

Example

To Locate Y0= 7U and the delta value of -4U, users need to input Y0=7U and delta Y = -4U

The waveform window will automatically locate the points and insert a tag with its delta X and delta Y value.

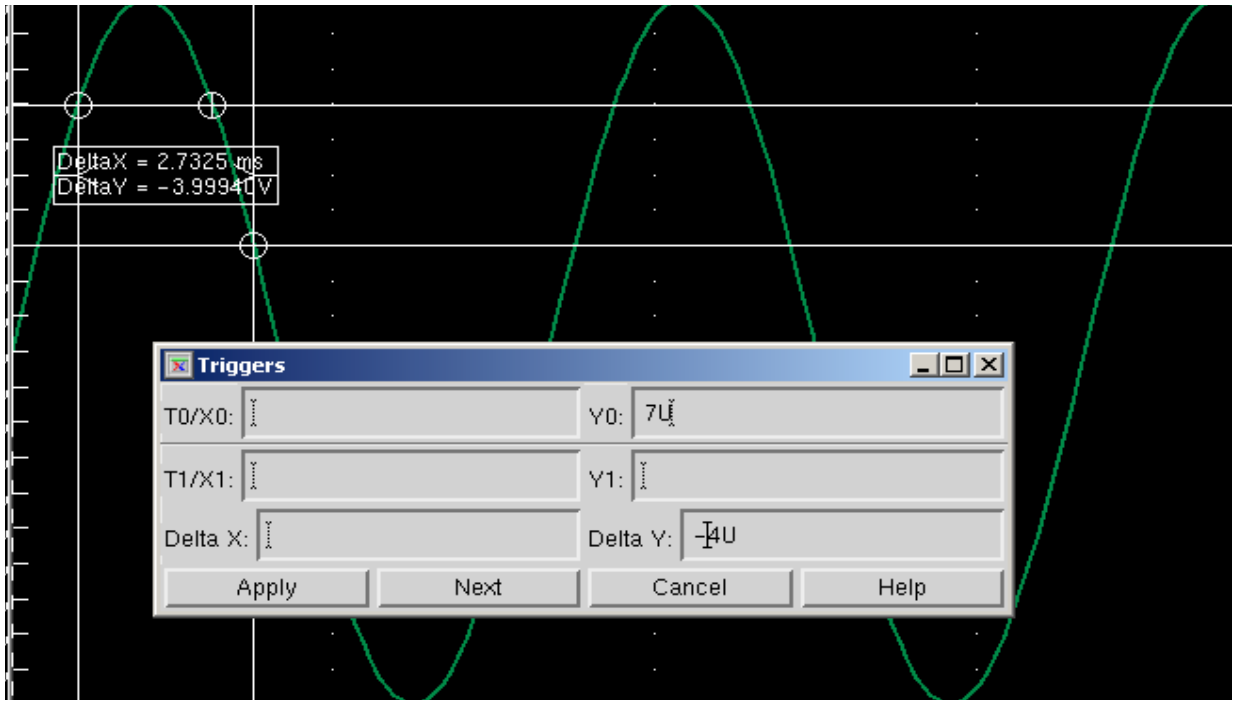


FIGURE 84. First set of data is triggered

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

By clicking next, user can trigger the next set of value .

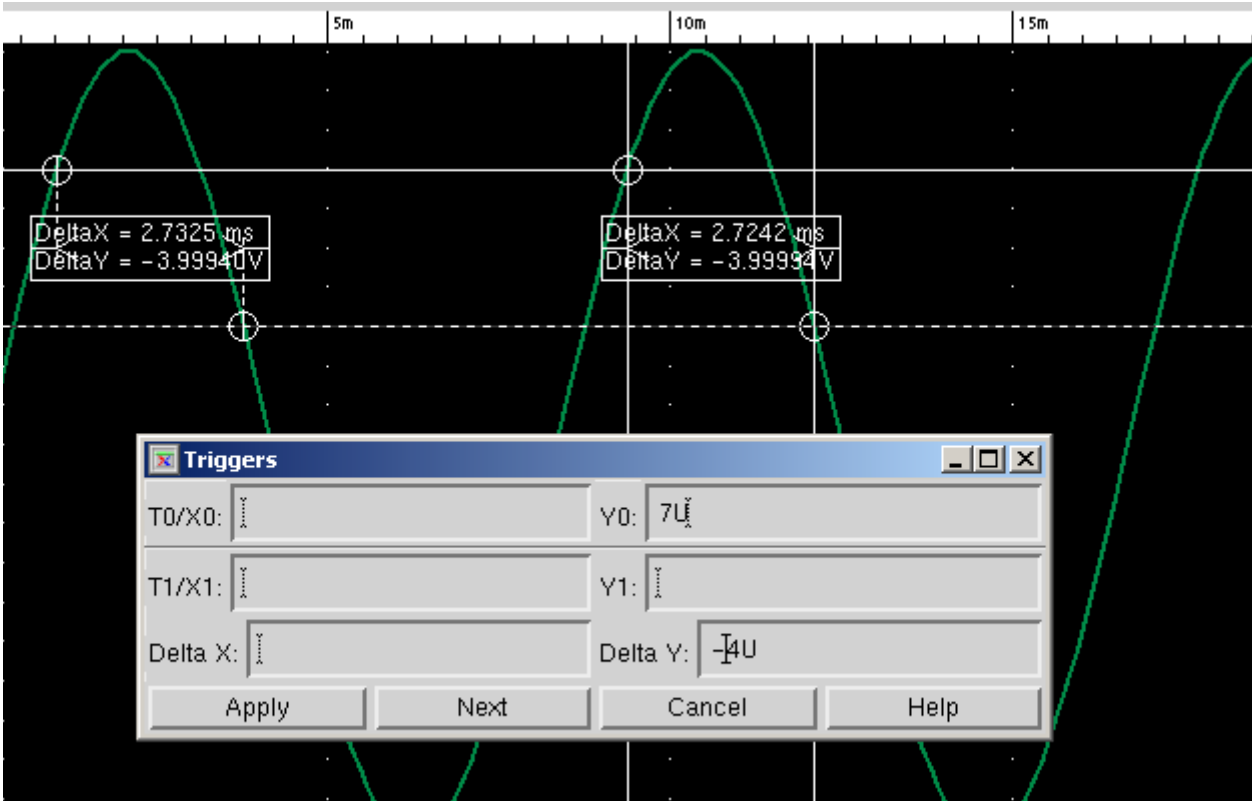


FIGURE 85. Next set of data is triggered by clicking next

By continuously clicking next, users are able to locate different sets of data with the same trigger criteria.

Analog Tools

This section describes the **Analog Features** in Undertow. Use the **Tools** pull down menu to select these features. This section describes *only* the analog features of the **Tools** pulldown menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

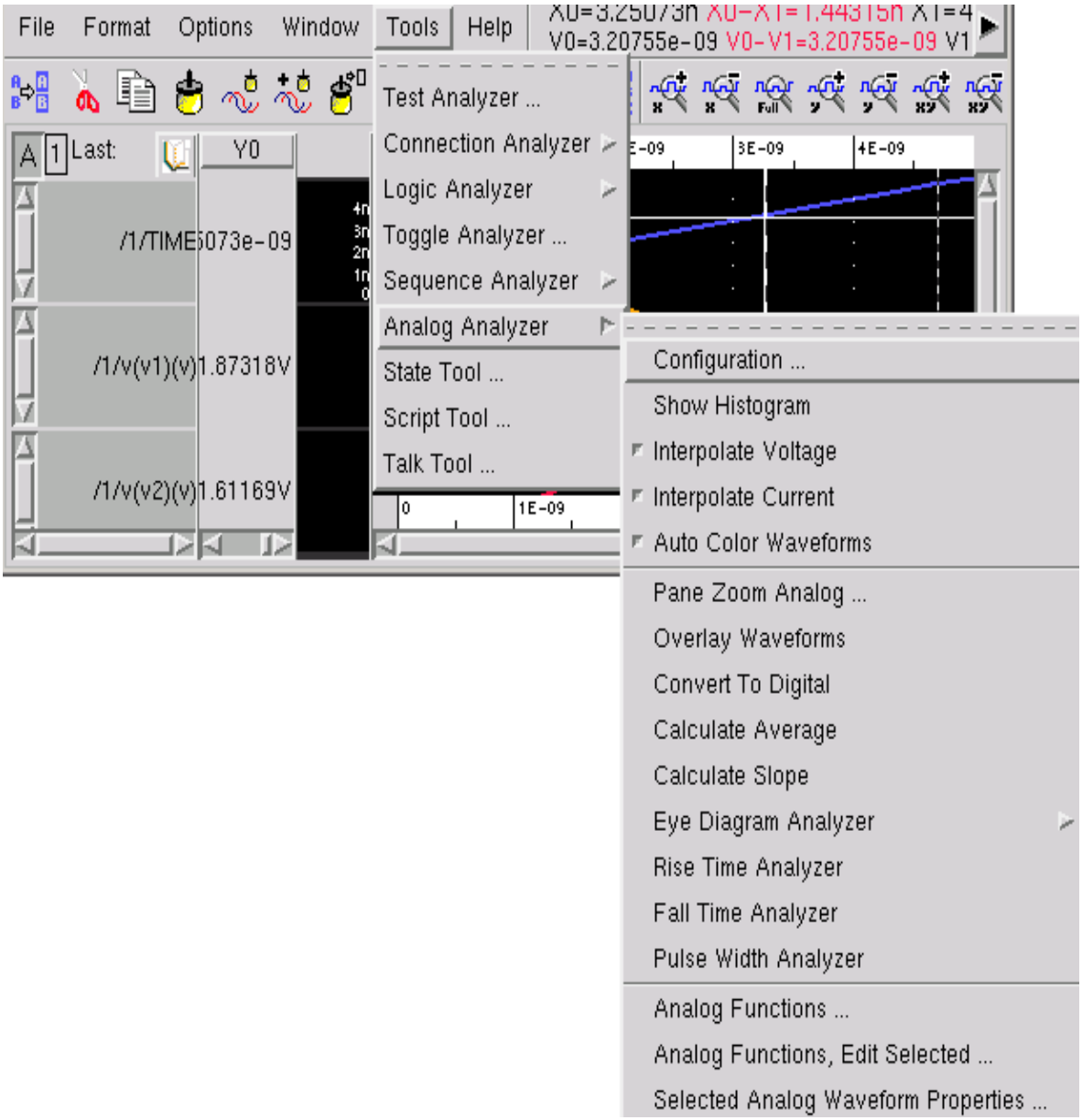


FIGURE 86. Analog Analyzer

Configuration

This option is the same as pressing the “Analog Configuration” icon.

Show Histogram

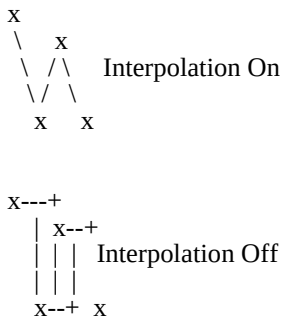
This option will display histogram of the signal next to the signal name.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Interpolate Voltage and Interpolate Current

Toggles between **Interpolate** and **Don't Interpolate**.

- If you select to **Interpolate** signal values, Undertow uses straight lines to connect all points on the analog signal. This might result in values between points, that are not in the original data file.
- If you select **Don't Interpolate**, Undertow uses only horizontal or vertical straight lines to connect all points. All values of intermediate points are equal to data that is already in the waveform data file.



Auto Color Waveforms

This option will let Undertow assign different colors each waveforms.

Pane Zoom Analog

This option is the same as selecting ***Window*** => ***Pane Control Pane*** => ***Zoom Time*** from the menus.

Overlay Waveforms

This option opens the **Overlay Analog Signals** window, which you can use to overlay analog waveforms.

1. Paste the signal names to superimpose in the editing area.
2. Click **Add** to display the signals.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

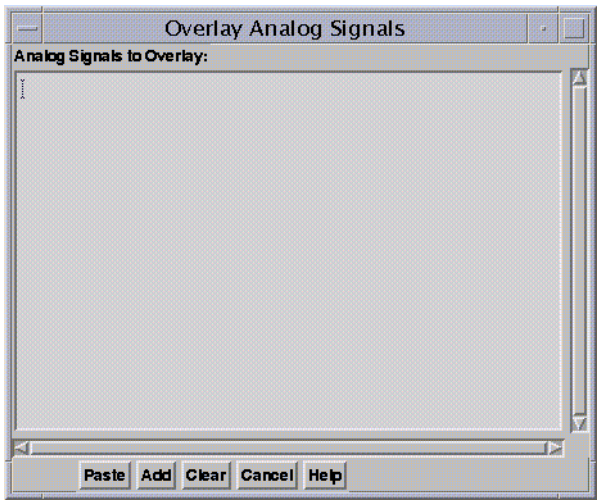


FIGURE 87. Overlay Analog Signals

Convert To Digital

This option converts analog signals to digital.

Use **Append to Names** to modify the selected signal names, after you convert them to digital. Threshold units are in (V)olts.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

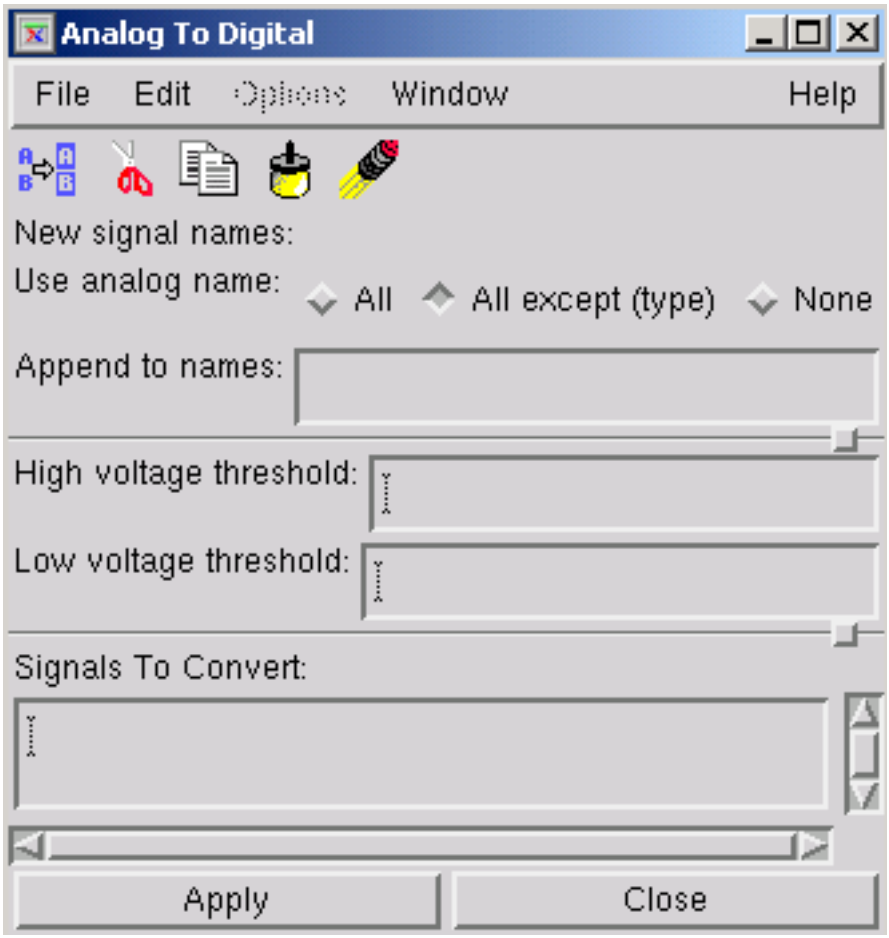


FIGURE 88. Analog To Digital

Calculate Average

This option displays the average value (0 to 459100) of the selected signals, for the time window bordered by the **T0** and **T1** time cursors.

Calculate Slope

Displays the slope (0 to 459100) of the selected signals, between the intersection points of the **T0** and **T1** time cursors and the waveforms.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Eye Diagram Analyzer

This option opens the **Eye Diagram Signals** window. You can use this window to overlay a single analog waveform onto itself, shifted by the fold-over time that you specify.

1. Select **Create Eye Diagram**.

   The **Eye Diagram Signals** window opens. Use this window to overlay a single analog waveform onto itself, shifted by the *fold-over* time.

2. Select the signal name to use as the basis of the eye diagram.

   The signal area automatically lists the selected signal.

3. To choose a color for the diagram, select a color name from the list of colors in the dialog.

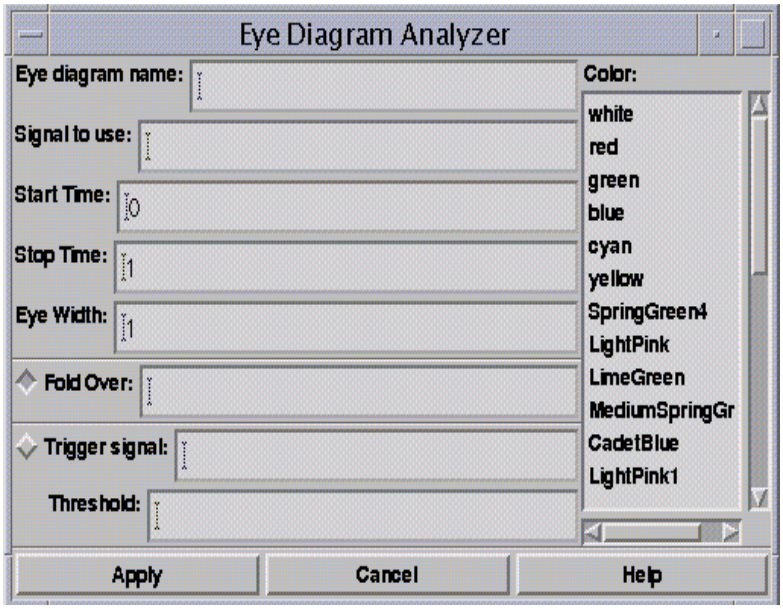


FIGURE 89. Eye Diagram Analyzer

4. In the **Start Time** field, indicate the starting time for the eye diagram.
5. In the **Stop Time** field, indicate the time after which no further overlaying occurs.
6. In the **Foldover Time** field, indicate the amount to shift each overlay (of the same signal) forward in time.
7. Click **Apply** to display the eye diagram.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Rise Time Analyzer

This option opens a window that you can use to calculate the time involved with the rising edges of an analog waveform.

1. Select **Rise Time Analyzer**.

   The **Rise Time Analyzer** window opens. Use this window to calculate the time involved with the rising edges of an analog waveform.

2. To place the signal in the **Selected Signal** area, highlight a waveform in the Undertow window.

3. Set the **Start Time** and **End Time**, either manually, or by placing cursor **0** at the beginning of the interval and cursor **1** at the end of the interval.

4. Use the **Low Thresh** and **High Thresh** fields to set the percentages (thresholds), between which Undertow measures the rise time.

   The RT table displays the results.

   - The first value is the time when Undertow measured the rise time.
   - The second value is the actual time that the waveform requires, to travel between the **Low** and **High** thresholds.
   - The third value is a number assigned to that specific rising edge.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

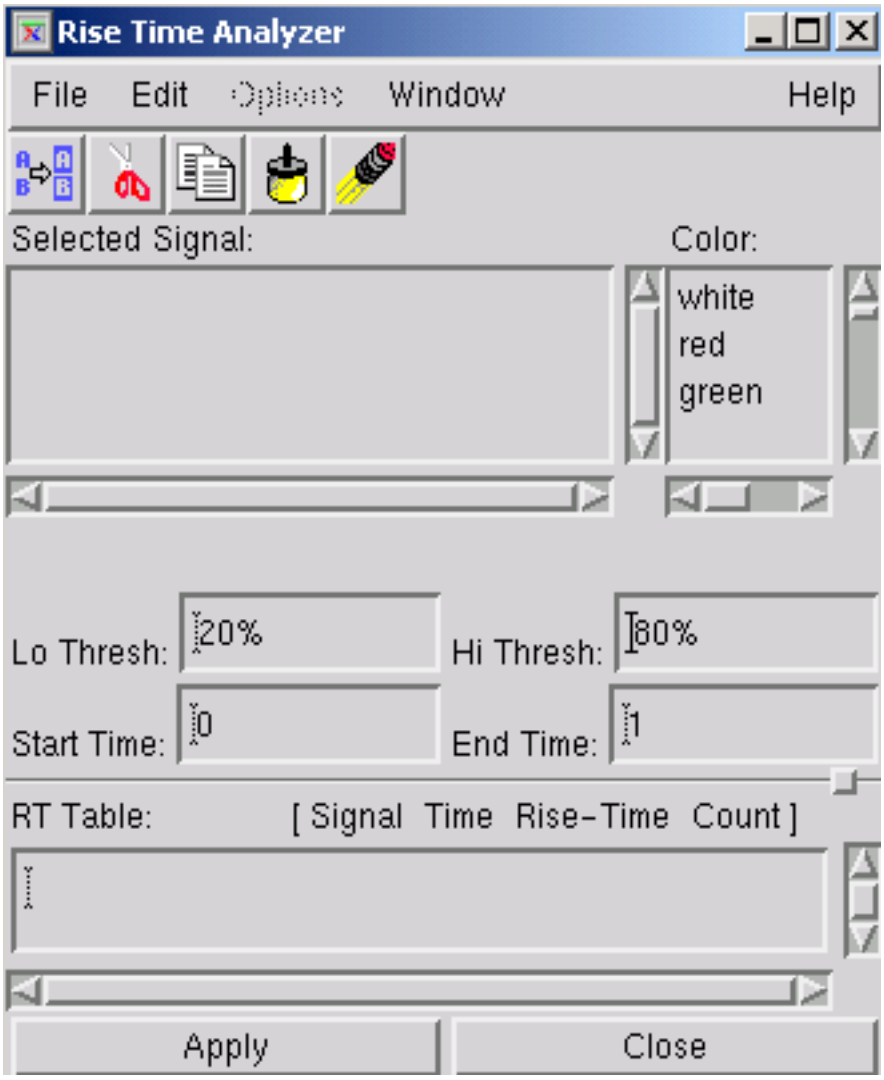


FIGURE 90. Rise Time Analyzer

Fall Time Analyzer

This option opens a window that you can use to calculate the time involved with the falling edges of an analog waveform.

1. Select **Fall Time Analyzer**.

   The **Fall Time Analyzer** window opens. Use this window to calculate the time involved with the falling edges of an analog waveform.

2. To place the signal in the **Selected Signal** area, highlight a waveform in the Undertow window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Set the **Start Time** and **End Time**, either manually, or by placing cursor **0** at the beginning of the interval and cursor **1** at the end of the interval.

4. Use the **Low Thresh** and **High Thresh** fields to set the percentages (thresholds), between which Undertow measures the fall time.

   The FT table displays the results.

   - The first value is the time when Undertow measured the fall time.
   - The second value is the actual time that the waveform requires, to travel between the **High** and **Low** thresholds.
   - The third value is a number assigned to that specific falling edge.

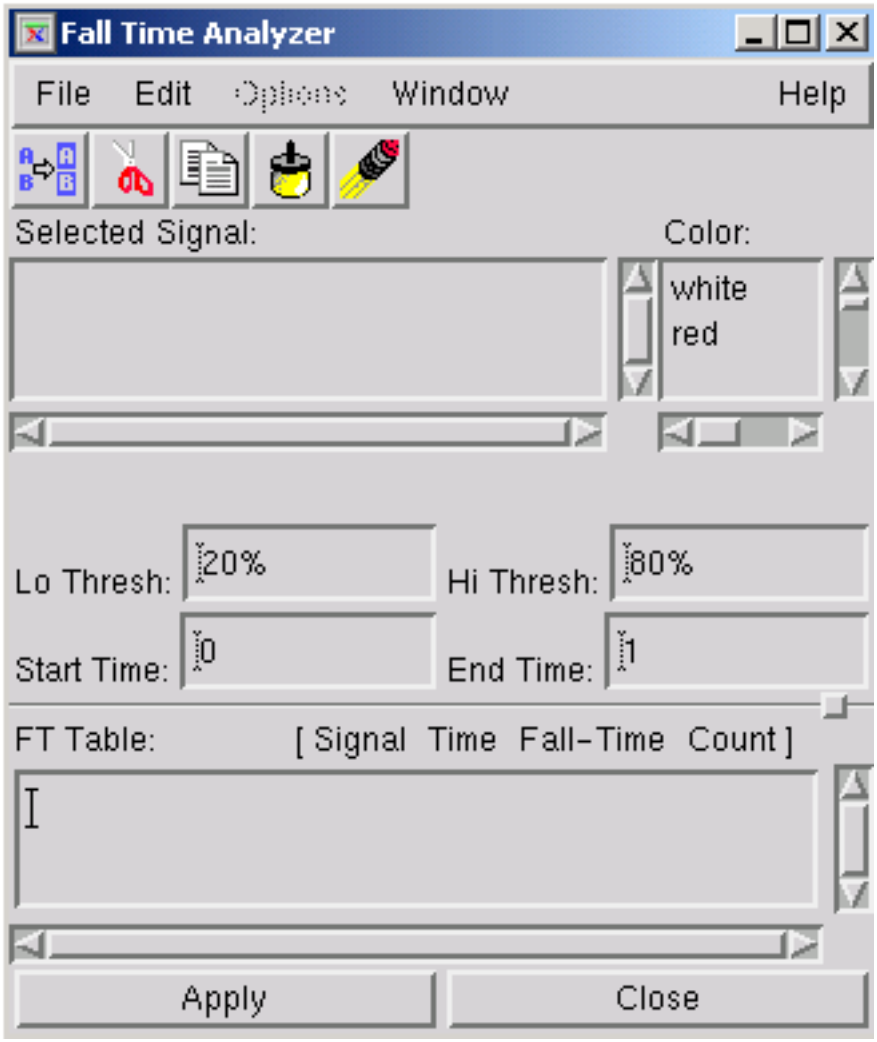


FIGURE 91. Fall Time Analyzer

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Pulse Width Analyzer

This option opens a window that you can use to calculate the time involved with pulses in an analog waveform.

1. Select **Pulse Width Analyzer**.

   The **Pulse Width Analyzer** window opens. Use this window to calculate the time involved with pulses in an analog waveform.

2. To place the signal in the **Selected Signal** area, highlight a waveform in the Undertow window.

3. Set the **Start Time** and **End Time**, either manually, or by placing cursor **0** at the beginning of the interval and cursor **1** at the end of the interval.

   Undertow tabulates pulse widths for waveforms that rise above the specified Threshold.

   - The first value in the PW table is the time when Undertow measured the pulse width.
   - The second value is the actual duration (time) of the specified pulse.
   - The third value is a number assigned to that specific pulse.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

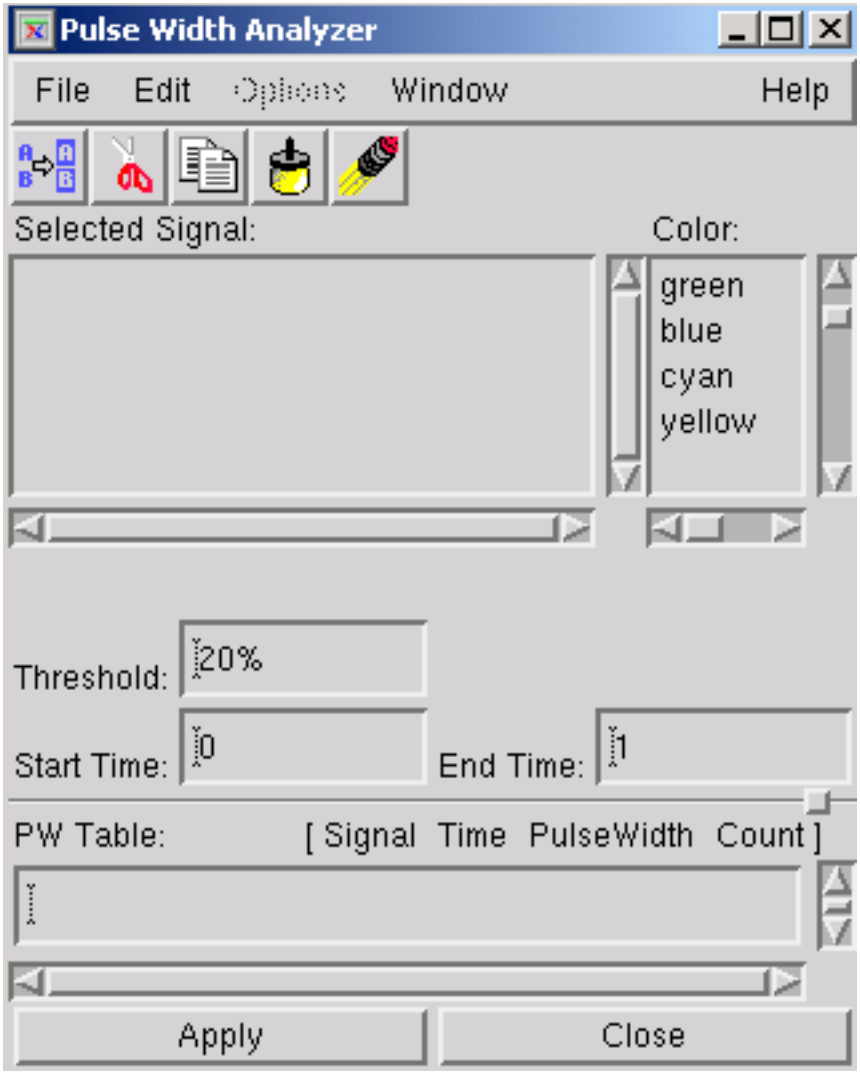


FIGURE 92. Pulse Width Analyzer Dialog

Analog Functions

This option opens the **Boolean Functions** window (Figure 93, “Analog Functions”).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

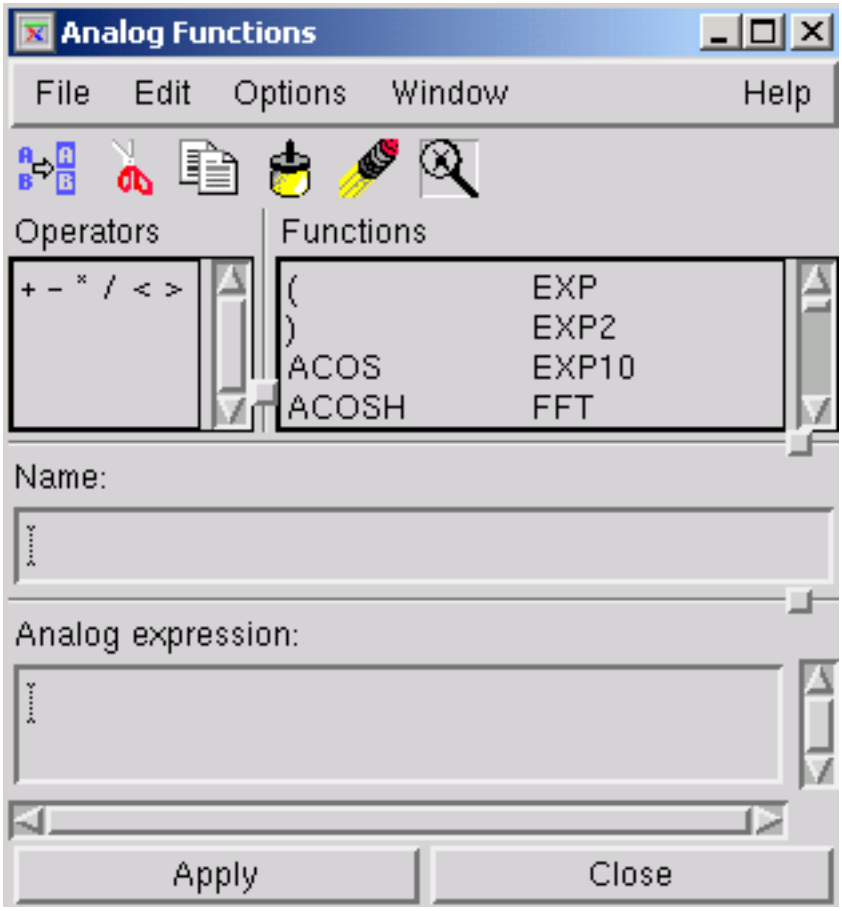


FIGURE 93. Analog Functions

This window includes the following areas:

- Operators used in creating analog expressions.
- Functions used in building analog expressions.
- A field in which you enter the signal name to which Undertow applies the expression.
- A window displaying the analog functions that you have built.

Analog Functions opens the Analog Functions window. This window has the following areas:

- Main Menu, at the top.
- Quick Command Bar, under the main menu.
- Operators and Functions lists.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Name input field.
- Analog Expression input area.
- Actions buttons:
  - **Apply** creates the new analog waveform, based on the current analog expression.
  - **Close** closes the **Analog Functions** window.

The main menu in this window has the following sections:

File—saves the current analog expression to a file, or loads a previously-saved expression. The **Print** option prints the analog expression in a readable format.

Edit—lets you select, copy, cut, or paste signals from the display window.

Window—lets you configure the current window. The configuration dialog box has two input fields:

- **Working Directory**—specifies which directory to save the setup files to, or load the setup files from.
- **File Suffix—**specifies the default suffix that the **File Selection** window uses.
- **Help**—opens the related help window.

You can use two radio buttons to switch the **Quick Command Bar** to either Glyphs (pictures) or Text.

- The **Apply** button enables the displayed settings.
- **Cancel** discards any modifications.

To construct an analog expression:

1. Paste signals into the Analog expressions area.
2. Combine these signals. You can either:
   - Use operators that you pick from the **Operators** list, or
   - In the **Name** input field, enter a name for the new signal. Then click **Apply** to apply functions to the analog signal.

For explanation of Filters:

1. Click on a the Filter that you want.

   An **Analog Filter Builder Window** opens.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Click on the Help button.

Examples:

1. Drag and drop a signal name (full hierarchy) into the **Analog Expression** window.
2. From the **Operators** list, select the < operator.
3. Drag and drop another signal name into the **Analog Expression** window.

Or:

1. Type a signal name in the **Name** input field.
2. Select a signal in the display.
3. Click the **Apply** button.

The new signal displays above the other (highlighted) signal. It is a digital signal that must have a value of either **0** or **1**.

*Note: **1** indicates that the value of the signal to the left of < is smaller than the signal on the right side of <.*

To compare analog signals:

1. To apply a math function to an analog signal, select the math function.

   For example, select SQRT (square root).

2. Select **(** from the **Functions** list.

Or:

1. Drag and drop a signal into the **Analog expression** window.
2. Select **)** from the **Functions** list.

After you complete either of these two procedures:

3. Type the name of the new signal in the Name field.
4. Click the **Apply** button.
5. Select another math function.

   For example, select *LOG10* from the functions list.

6. Drag and Drop a signal into the **Analog Expression** window.
7. Type the name of the new signal in the **Name** field.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

8. Click the **Apply** button.

   The **Waveform** window displays the new signal.

Other examples:

MAGN (FFT (signal, 1e9))
PHASE (FFT (signal , 1e9))

This example performs a Fast Fourier Transform (FFT) on the selected signal, to show the frequency components (up to a limit of 1e9 Hz, which is 1 Gigahertz). Place the results in an **Arbitrary X** pane.

The sampling frequency in this example is 1e9 Hz. Computation time increases 100X for every 10X increase in sampling frequency. This signal has a strong DC component (it is always positive), so to see the other frequency components in the MAGN plot, use ***Option*** **=>** ***Properties*** **=>** ***Selected Analog Waveform*** to set the Top Value to 0.3.

MAGN (IFFT (FFT (signal , 1e9) , 1e9))

This example performs an IFFT on the FFT above. The result is similar to running the signal though a lowpass filter. This plot should be nearly identical to the original signal, with a little smoothing. Place the results in a **Time** pane.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

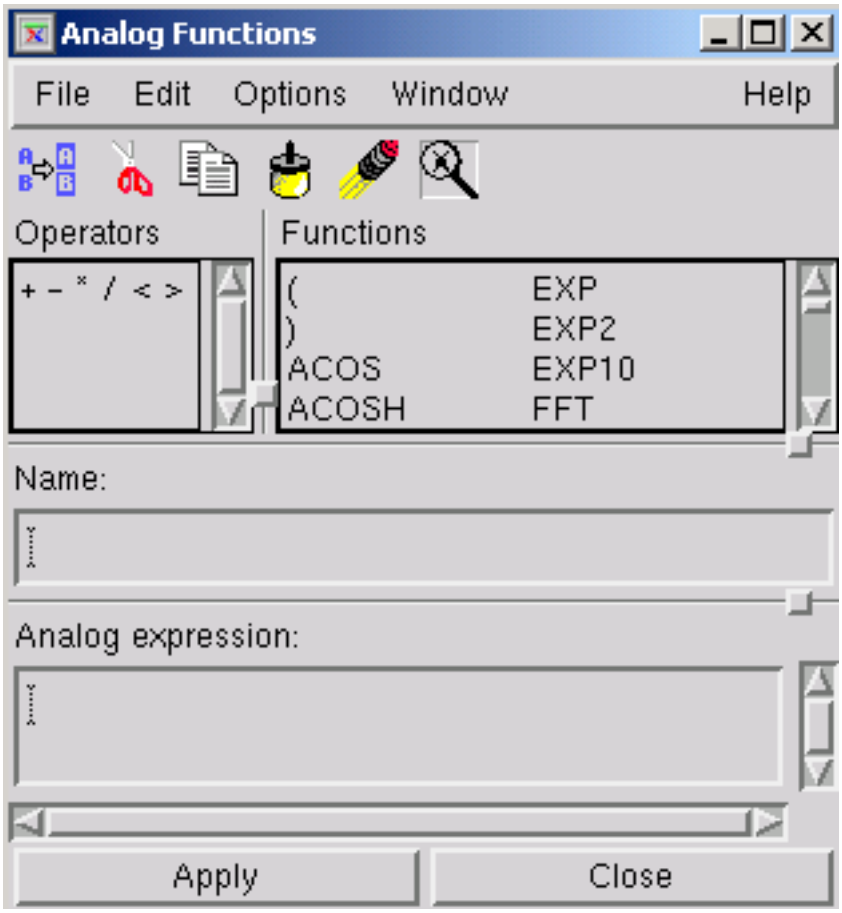


FIGURE 94. **Analog Functions Window**

Analog Functions, Edit Selected.

This function opens the **Boolean Functions** window, for a signal that you already selected.

Selected Analog Waveform Properties

This option opens the **Select Overlay Properties** window. You can use this window to set the properties of analog waveforms.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

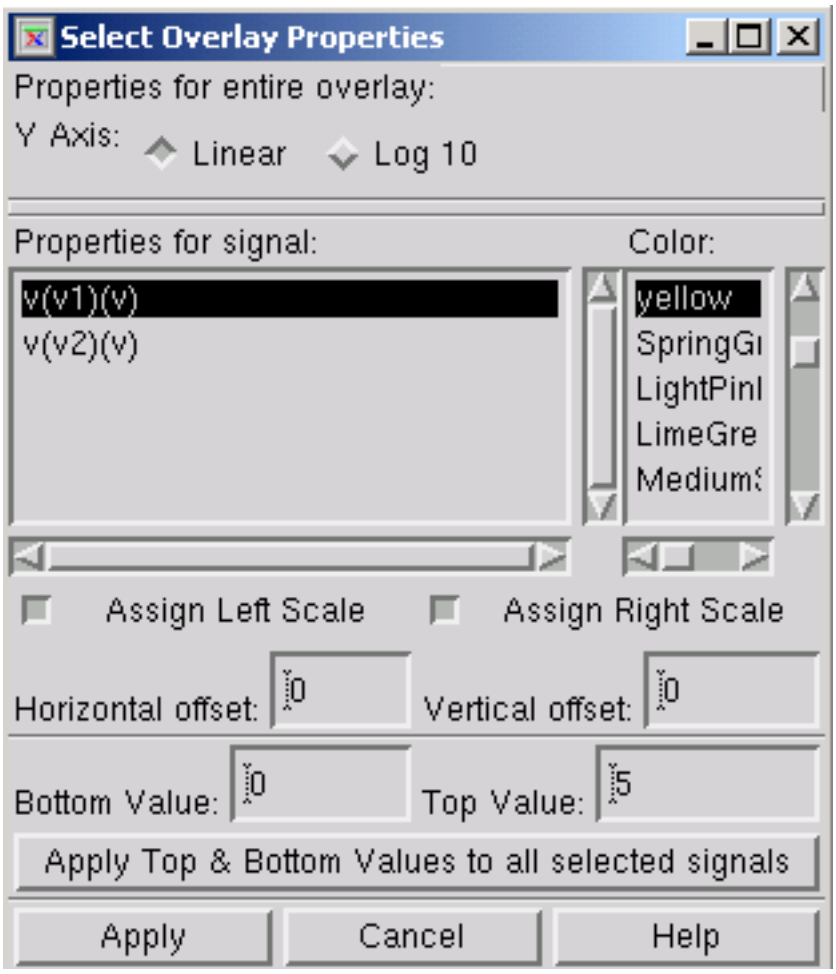


FIGURE 95. Select Overlay Properties

1. To set the properties of a single waveform, or of overlaying waveforms, open the **Selected Analog Waveform** window.
2. Highlight the waveform for which you want to set properties.

   The name of the selected waveform displays in the **Properties for Signal** box.

You can use this window to set the following properties:

- **Color**—You can choose from fifty colors. To set a color for any or all overlaying waveforms:
  1. Highlight the specific waveform in the **Properties for signal** box.
  2. Select a color.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Assign Scales**—You can assign two different scales for a set of overlaying waveforms, or you can set a single scale for each waveform. To make full use of the left and right scales for overlaying waveforms:

  1. Select **Options->Properties->Grids & Scale** from the menus.
  2. Select **Always** for **Show left and right scales**.

  You can then set two different scales and view them:

  1. To set the first scale (left), set the bottom and top values.
  2. Make sure that the correct waveform(s) are highlighted, and click **Apply**.
  3. Click the **Assign Right Scale** button.
  4. Repeat this process.

- **Horizontal and Vertical Offset**—You can set these offsets to either scale.

  1. Use **Vertical offset** to offset the voltage and current by the specified amount.
  2. Use **Horizontal offset** to offset the time by the specified amount.

- **Properties for Entire Overlay**—Whether the waveform shows voltage, current, or another property. you can display the waveform as a linear or base 10 log plot. To set this option, click the correct button for the property that you want to set, and then click **Apply**.

Script Tool

This option opens the **Script Tool** window. For a more detailed description, see Section , “Script Tool”.

Command Line

To open the Command Line Interface, at the bottom of your Undertow Window:

1. Click on the **Analog Configuration** icon.
2. Turn *ON* the **Command Line Interface**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

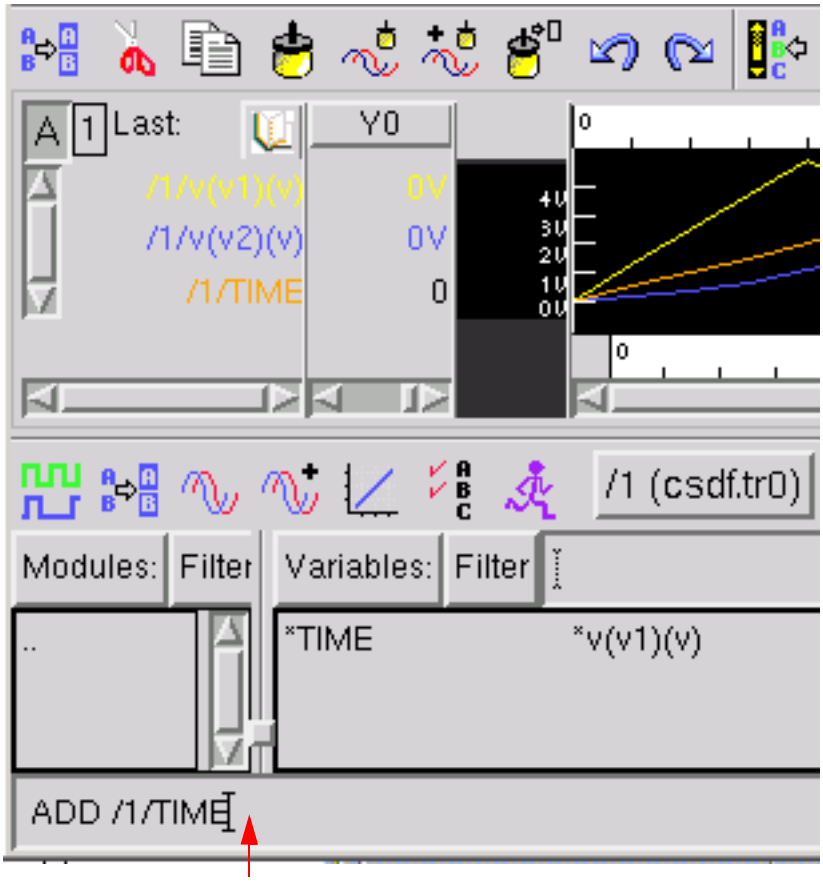


FIGURE 96. Command Line Interface

TABLE 41. Interface Command

Command	Description	Usage/Examples
a	ADD signal to existing y-axis	**Add v1 v2 v1+v2** plots the three traces: • v1 • v2 • v1+v2 on the same y-axis.
c	CLOSE	Closes all open files and deletes all.
d a	DEL ALL	**DEL ALL** deletes all signals, deletes the x and y axes, and clears the screen, but it keeps the file.
d x	DEL X	**DEL X** deletes the selected pane (x-axis).
d x#	DEL X#	**DEL X#** deletes the specified pane (x-axis).
d y	DEL Y	**DEL Y** deletes the selected signals (y-axis).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 41. Interface Command (Continued)

Command	Description	Usage/Examples
d y#	DEL Y#	**DEL y#** deletes the specified overlay (y-axis).
e	EXIT ut	Exits Undertow.
f	FREE (clear)	Frees (clears) the command line history.
h	HELP	Displays help information for the command line interface.
li	LIST (filter_expression) shows chooser with optional variable filter.	**LI** lists all signals (urns on the **Chooser**). **L*v1** lists all signal names that end in **v1**. **LM** lists macro definitions. **LOFF** turns off the list (**Chooser**) window.
lm	LMACRO	Displays a list of defined macros.
lo	LOFF	Closes the **Chooser**.
m	MACRO	*macro_name*(*arguments*)=*expression*=define macro • **m myfunc(cow,pig)=2*cow+pig+v1** creates a new function of two arguments (**cow** and **pig** serve as placeholders). Subsequently, entering: **a myfunc(v2, v3)** plots the waveform of **2*v2+v3+v1**, but the label is: **myfunc(v2, v3)** • **m mysum+v1+v2+v3** creates an alias without arguments, for **v1+v2+v3**. *Note: Do not place spaces between parentheses, values, or arithmetic operators.* Order of operations is: 1. multiplication. 2. division. 3. addition. 4. subtraction.
n	NEW signal	Displays a signal in a signal new overlay (y-axis).
o	OPEN filename	Opens the specified files, matching **regex**. You can open multiple files at once—for example: **open *.ac0** opens all AC files with a **.ac0** file extension.
pa	PAUSE	Waits for user input.
pr	PRINT	Print the selected signals.
q	QUIT	Exits Undertow.
r	RUN filename	Runs the specified command-line script file.
s	SIGNALS signal	Displays signals in separate new overlays (y-axis)
w	WRITE filename	Writes the command-line history to the specified file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 41. Interface Command (Continued)

Command	Description	Usage/Examples
x#	X#	Selects the specified pane (x-axis).
xli	XLIN	Sets the selected pane (x-axis) to **linear** scale.
xlo	XLOG	Sets the selected pane (x-axis) to **log** scale.
XVAR	set x-axis variable	**XVAR v2** sets the x-axis variable value to **v2**.
y#	Y# selects y-axis	**Y5** selects the fifth y-axis of the active x-axis. If there are fewer than five y-axes, then Undertow generates new y-axes until there are 5, and selects the fifth y-axis.
yli	YLIN	Sets the selected y-axis to **linear** scale.
ylo	YLOG	Sets the selected y-axis to **log** scale.
Yr	YRANGE	Sets the bottom and top values for the selected overlay (y-axis).
?	?	Displays help information for the command line interface.

Functions Calculator

1. From the pulldown menus, select ***Tools* => *Analog Analyzer* => *Analog Functions***.

   The **Analog Function Calculator** opens.

   *Note: All analog functions require only a signal name for input, except **MAGN (absolute_value)**.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

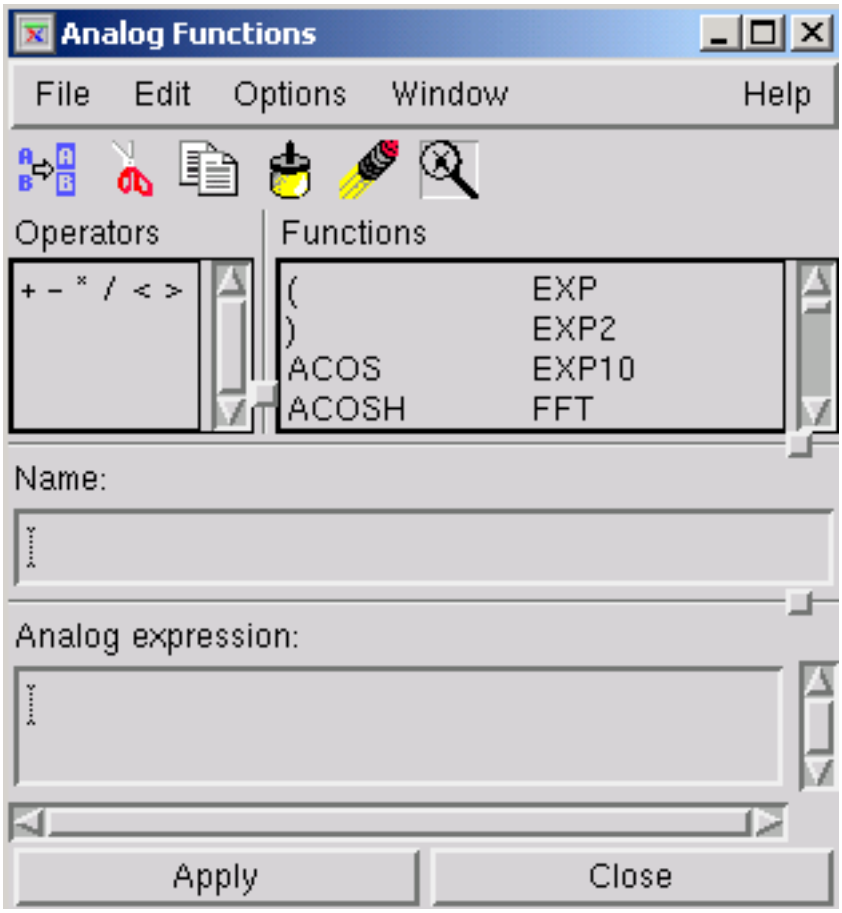


FIGURE 97. Analog Functions Window

The **Functions** window lists all available functions.

2. To construct an analog expression, pass signals into the Analog expressions area.
3. To combine the signals, either:
   - Select Operators from the Operators list, or
   - Apply functions to the analog signal.
4. In the **Waveform Chooser** window, press and hold the middle mouse button on the signal to select.
5. Drag the selected signal, and drop it into the **Analog Functions** window.
6. To clean the functions written in the lower pane, click the **Clear** icon.

   This is the fifth icon from the left, and resembles a pencil that is erasing.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

When the **Analog Functions** window completes a calculation, Undertow creates a new signal.

1. To find this signal, use the button in the right pane of the **Waveform Chooser**.

   The waveform display moves up a level in the hierarchy of the signals.

2. Move up to the highest level.

   Undertow lists the different inputs files that are currently loaded. The right pane lists all new signals that you created.

Note: You can use these created signals as inputs for further calculations.

TABLE 42. Analog Functions

Name	Description
ACOS	arc cosine function
ACOSH	hyperbolic arc cosine function
ACOSPI	inverse hyperbolic arc cosine function
ASIN	arc sine function
ASINH	hyperbolic sine function
ASINPI	inverse hyperbolic sine function
ATAN	arc tangent function
ATANH	hyperbolic arc tangent function
ATANPI	inverse hyperbolic tangent function
AVERAGE	average function
BESSEL	Bessel function
BUTTERWORTH	Butterworth function
CBRT	compute cube root
CHEBYSHEV1	Chebyshev polynomial of the first kind
CHEBYSHEV2	Chebyshev polynomial of the second kind
COEFPS	coefficient

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 42. Analog Functions (Continued)

Name	Description
COMPLEX	complex function Example: **COMPLEX (*signal_name*{real}(v), *signal_name*{imag}(v))** • The first argument is the data to use for the **real** values. • The second argument is the data to use for the **imaginary** values. Typically, you use this function to put together complex signals from real and imaginary *y* parts of an ac or dc **CSDF** file format.
COS	cosine
COSH	hyperbolic cosine
DB10	Decibel (10*LOG())
DB20	Decibel (20*LOG())
DIFF	difference function
ELLIPTIC	elliptic curve function
ERF	error function
ERFC	complementary error function
EXP	exponential function
EXP2	base 2 exponential function
EXP10	base 10 exponential function
FFT	Fast Fourier Transform Examples: **FFT(*data, sampling_frequency*)** **FFT(*data, sampling_frequency, start_time*)** **FFT(*data, sampling_frequency, start_time, end_time*)** *data* is usually a signal name, but it can be any data such as (*signal_name1* + s*ignal_name2*), which is why this example uses the variable name ***data*** instead of ***signal_name***.
FIR	Finite Impulse Response Filter
IFFT	Interpolated FIR filter
IIR	Infinite Impulse Response Filter
IMAG	Extracts the imaginary portion of a complex-valued expression, which can have a positive or negative value. IMAG is one of four functions for extracting real-valued data from complex-valued data (the other three are MAGN, PHASE, and REAL).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 42. Analog Functions (Continued)

Name	Description
INTEG	Integral
LOG	log function
LOG2	binary (base 2) log function
MAGN (*absolute_value*)	Extracts the magnitude portion of a complex-valued expression, which can have a positive or negative value. MAGN() is one of four functions for extracting real-valued data from complex-valued data (the other three are PHASE, REAL, and IMAG).
MAX	minimum of a list
MIN	maximum of a list
PHASE	Extracts the phase portion of a complex-valued expression, which can have a positive or negative value. PHASE is one of four functions for extracting real-valued data from complex-valued data (the other three are MAGN, REAL, and IMAG).
PLOTXY	plotxy function
POW	power function
REAL	Extracts the real portion of a complex-valued expression, which can have a positive or negative value. REAL is one of four functions for extracting real-valued data from complex-valued data (the other three are MAGN, PHASE, and IMAG).
RMS	• squares the signal • integrate over time • calculate the root of time average
SIN	sine function
SINPI	inverse sine function
SQRT	square function
TAN	tangent function
TANH	hyperbolic tangent
TANPI	inverse tangent function

Example 1:

1. Drag and drop a signal name into the **Analog Expression** field.
2. From the **Operators** list, select the < (less than) operator.
3. Drag and drop another signal name into the **Analog Expression** window.
4. Type a signal name in the **Name** input field.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

5. Select a signal in the display.
6. Click **Apply**.

   Undertow displays the new signal above the other (highlighted) signal.
7. Compare the two analog signals.
8. To apply a math function to an analog signal, select the math function (for example, **SQRT**), and select **()** from the **Functions** list.
9. Drag and drop a signal into the **Analog Expression** field, and select **)** from the **Functions** list.
10. Type the name of the new signal in the **Name** field.
11. Click **Apply**.

Example 2:

1. Drag and drop a signal name (full hierarchy) into the **Analog Expression** window.
2. From the **Operations** list, select the < (less than) operator.
3. Drag and drop another signal name into the **Analog Expression** window.
4. Type a signal name in the **Name** input field.
5. Select a signal in the display.
6. Click **Apply**.

   Undertow displays the new signal above the other (highlighted) signal. This is a digital signal, with values of **0** or **1**.

   - **1** indicates that the value of the signal to the left of < is smaller than the signal on the right side of <
   - **0** indicates that the value of the signal to the left of < is larger than the signal on the right side of <
7. Compare the two analog signals.

Example 3:

1. To apply a math function to an analog signal, select the math function (for example, **SQRT**), and select **(** from the **Functions** list.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Drag and drop a signal into the **Analog expression** window.
3. Select **)** from the **Functions** list.
4. Type the name of the new signal in the **Name** field.
5. Click **Apply**.

Example 4:

1. Execute the following command from the command line:

   MAGN(FFT(signal, le9)) and PHASE(FFT(SIGNAL, le9))

   This command perform a Fast Fourier Transform (FFT) on the signal, to show the frequency components (up to a limit of le9 hz (1 Gigahertz)).

2. Place the result in an **Arbitrary X** pane.
   - The sampling frequency is le9 hz.
   - Computation time increases by 100% for every 10X increase in sampling frequency.
   - This signal has a strong DC component (it is always positive).
3. To see the other frequency components in the MAGN plot, use ***Option*** => ***Properties*** => ***Selected Analog Waveform*** to set the *Top Value* to **0.3**.
4. Execute the following command from the command line:

   MAGN (IFFT (FFT (signal, le9), le9))

   This command performs an IFFT on the FFT above. (This is similar to running the signal through a lowpass filter).

   Note: This plot should be nearly identical to the original signal, with a little smoothing.

5. Place this result in a **Time** pane.

Analog Filter Builder Guide

Analog Filter Functions

- BESSEL
- BUTTERWORTH

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- CHEBYSHEV1
- CHEBYSHEV2
- ELLIPTIC

FILTER Syntax and Examples

For information about Parameters and their domains, refer to Table 43, “BUTTERWORTH, CHEBYSHEV1, CHEBVSHEV2, ELLIPTIC parameters description”.

- BUTTERWORTH

  BUTTERWORTH (*input_signal*, *filter_type*, *filter_order*, *pass_limit1*, *pass_limit2*, *sample_freq*)

  Example:

  BUTTERWORTH (input_signal , 1 , 6 , 0 , 2e8 , 1e10)

  This examples performs a lowpass filter on the signal.

  - Filter order is **6**.
  - Pass frequency is **2e8 hz** (200 Megahertz).
  - Sampling frequency is **1e10 hz** (10 Gigahertz).

- CHEBYSHEV1

  CHEBYSHEV1 (*input_signal*, *filter_type*, *filter_order*, *pass_limit1*, *pass_limit2*, *pass_ripple*, *sample_freq*)

  Example:

  CHEBYSHEV1 (input_signal, 2 , 4 , 30 , 0 , 5 , 100)

- CHEBYSHEV2

  CHEBYSHEV2 (*input_signal*, *filter_type*, *filter_order*, *pass_limit1*, *pass_limit2*, *pass_ripple*, *sample_freq*)

  Example:

  CHEBYSHEV2 (input_signal, 2 , 4 , 30 , 0 , 5 , 100)

- ELLIPTIC

  ELLIPTIC (*input_signal*, *filter_type*, *filter_order*, *pass_limit1*, *pass_limit2*, *stop_limit*, *pass_ripple*, *sample_freq*)

  Example:

  ELLIPTIC (signal , 2 , 6 , 2e8 , 1e30 , -60 , 0.5 , 1e10)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This example performs a highpass filter on the signal.

- Filter order is **6**.
- Pass frequency is **2e8 hz** (200 Megahertz).
- Sampling frequency is **1e10 hz** (10 Gigahertz).

This filter shows all of the high-frequency jitters in the signal; the lowpass showed the smoother, low-frequency elements.

TABLE 43. BUTTERWORTH, CHEBYSHEV1, CHEBVSHEV2, ELLIPTIC parameters description

Parameter	Description
input_signal	Input waveform, to pass through the filter.
filter_type	1=low-pass, 2=high-pass, 3=band-pass, 4=band-stop.
filter_order	Polynomial order of the filter.
pass_limit1	First edge of the pass band, in Hertz.
pass_limit2	Second edge of the pass band, in Hertz (set to **0** for low-pass and high-pass filters).
stop_limit	First edge of the stop band, in Hertz. Or if negative, minimum decibels (dbs) of attenuation for the stop band.
pass_ripple	Peak-to-peak ripple in the pass band, in dbs.
sample_freq	Sampling frequency of the filter, in Hertz.

- IIR

  IIR (*input_signal* , *input_coeff_list* , *output_coeff_list* , *sample_freq*)

TABLE 44. IIR parameters description

Parameter	Description
input_signal	Input waveform, to pass through the filter.
input_coeff_list	Coefficients to apply to the inputs. (Use the COEFFS function to build a list of coefficients).
output_coeff_list	Coefficients to apply to the outputs.
sample_freq	Sampling frequency of the filter, in Hertz

To implement the IIR filters, Undertow feeds back a weighted sum of past output values, and adds this sum to a weighted sum of the previous and current input values.

- FIR

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIR (*input_signal* , *input_coeff_list* , *sample_freq*)

TABLE 45. FIR parameters description

Parameter	Description
input_signal	Input waveform, to pass through the filter.
output_coeff_list	Coefficients, to apply to the outputs.
sample_freq	Sampling frequency of the filter, in Hertz.

- MAGN, PHASE

  MAGN (FFT (signal , 1e9))

  and

  PHASE (FFT (signal , 1e9))

  This example performs an FFT on the signal, to show the frequency components (up to a limit of 1e9 hz (1 Gigahertz)).

  Place the result in an **Arbitrary X** pane.

  - The sampling frequency is le9 hz.
  - Computation time increases by 100% for every 10X increase in sampling frequency.
  - This signal has a strong DC component (it is always positive).

  To see the other frequency components in the MAGN plot, use ***Option*** **=>** ***Properties*** **=>** ***Selected Analog Waveform*** to set the *Top Value* to **0.3**.

  Example:

  MAGN (IFFT (FFT (signal, le9), le9))

  This command performs an IFFT on the FFT above. (This is similar to running the signal through a lowpass filter).

  Note: This plot should be nearly identical to the original signal, with a little smoothing.

  Place this result in a **Time** pane.

  The other arguments to the **ELLIPTIC** filter are:

  - Passband ripple of 0.5 db.
  - Stopband attenuation of 60 db.

  Both filter results go into a **Time** pane.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Script Tool

You can use the Undertow Script Tool to create and run your own Perl scripts.

Window Description

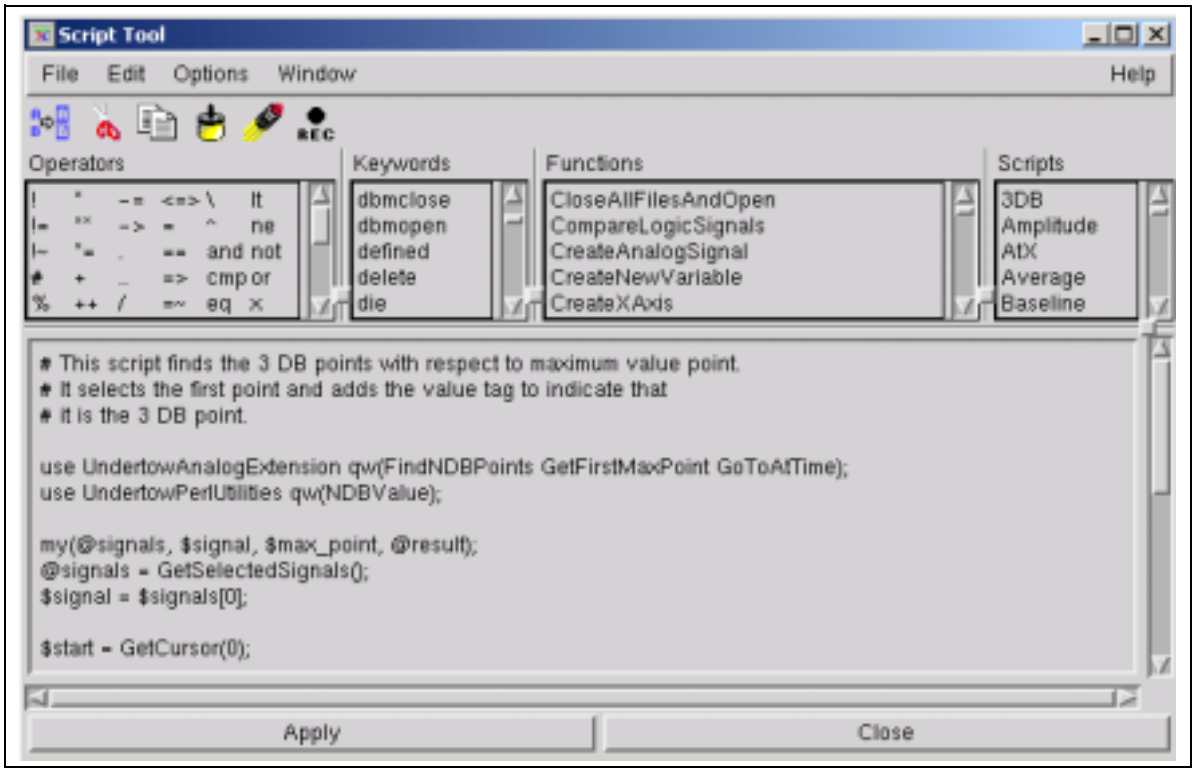


FIGURE 98. Script Tool Window

In the list boxes, Undertow provides all of the operators and keywords for the Perl language. These boxes also provide Undertow-specific functions, and a list of the scripts in your **scripts** directory.

Menus

The following menus are part of the Script Tool Window:

- File - Contains **Open..**, **Save**, **Save as**, **Print**, **Close**.
- Edit - Contains **Select All**, **Cut**, **Copy**, **Paste**, **Clear**.
- Options - Contains **Record**.
- Window - **Configure**, **Show Selector**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

*Note: Use **Window=>Configure=>**Working Directory field to set the location of the scripts directory. usually ut8.3.3/scripts. By default the Working Directory field is set to the UT_WORK_DIR, which may be configured differently.*

Buttons

The toolbar contains the **Select All**, **Cut**, **Copy**, **Paste**, **Clear** buttons. The window buttons are **Apply** and **Close.**

Fields

For more information on using Perl to build scripts refer to . Choose a script, or build a script from the following field:

- Operators.
- Keywords.
- Functions.
- Scripts.

Builder, a graphical Perl window building tool

The script tool not only builds textual Perl scripts that can interact with Undertow, but also includes a graphical window builder for Perl, called *Builder*.

Note: To ensure that scripts run properly, place the images *subdirectory under* scripts.

You can use the script tool to write scripts, and then use *Builder* to build a graphical window that facilitates the execution of this script. This graphical window can contain:

- Text areas for you to enter input.
- Text areas for Undertow to output the results.
- Buttons, onto which you can attach scripts.
- Other graphical elements to make running your scripts easier.

You can use Builder to create a *template* (a partial graphical window), and save it in the **Templates** window. You can then use the template as a basis for building other windows, which simplifies the building of windows. You can build templates, and then use them repeatedly to create many different windows.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

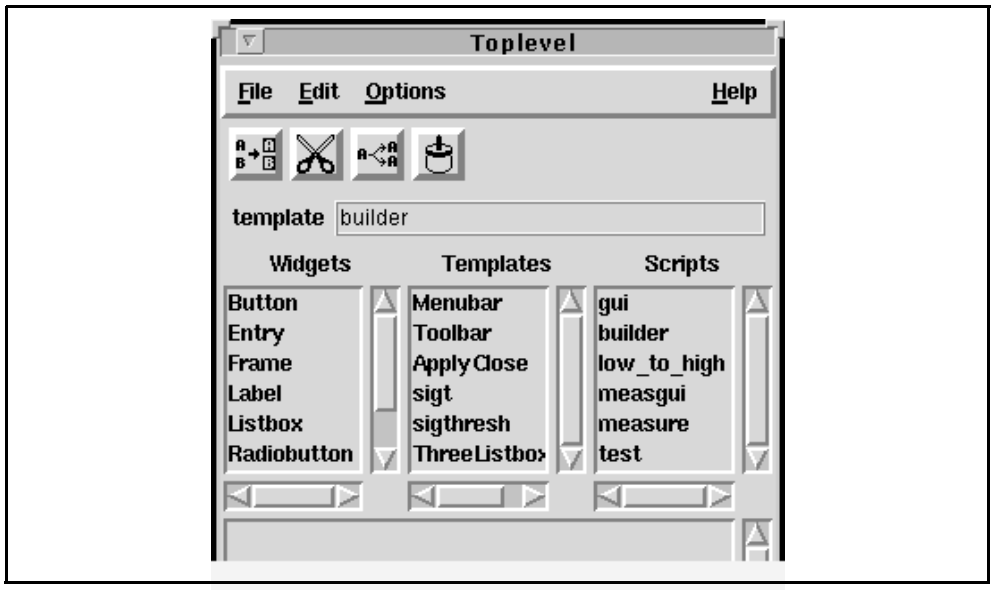

Toplevel
File Edit Options Help
template builder
Widgets Templates Scripts
Button Entry Frame Label Listbox Radiobutton
Menubar Toolbar Apply Close sigt sigthresh ThreeListbox
gui builder low_to_high measgui measure test

FIGURE 99. Builder Window

The Builder window, as shown in Figure 99, “Builder Window”, has three sections:

- Widgets.
- Templates.
- Scripts.

Widgets

The **widgets** window contains low-level widgets that you can use to build templates or windows. When you select any widget, Undertow inserts it onto a window frame. Use the middle mouse button to reposition or move the widget.

These widgets are:

- **Buttons**—Pops up the **Button** widget. This widget adds push buttons to the window.
- **Entry**—Pops up an **Entry** widget. Use entry widgets to type text, drag and drop text, or display text results.
- **Frame**—Serves as geometric frame. The separator is made from the frame widget, but is changed into a thick line.
- **Label**—A text label that you can use to label areas in the window.
- **List Box**—This widget can hold a text list.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Radio Button**—This widget holds a button and a text label.
- **Scroll Bar**—Pops up the scroll bar widget.
- **Text**—Pops up the text widget, which you can use to enter or display text.

After you place a widget onto a window, you can move it to any position. When you press the right mouse button over a widget, you can select characteristics attached to the selected widget. The following sections describe these widget characteristics.

Command

This option opens a window displaying the current command that is attached to this widget. To attach a new command to the widget:

1. Enter a command in the **New Command** window.
2. Click Apply.
3. Click Close.

The new command attaches to this widget. Commands are generally names of Perl subroutines. To attach a Perl script, instead of a subroutine, enter the following in the **New Command Window**:

```
&DoScript("Script_name");
```

NOTE: The &DoScript("Script_name"); causes the script_name to be executed.

Configure

This option opens the configuration for this widget. The configuration for the widget is displayed, and windows show both the current value and the new value for the configuration parameter.

1. Press on any parameter in the right list box to see its current value.
2. To change the size of a widget, press the **Shift** key, and then press the middle mouse button on any widget.
3. To stretch or shrink the size of this widget, drag the mouse button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Destroy

This option removes the widget.

Flash

This option flashes a specific widget, so you know that the cursor is over this particular widget.

Text

You can use this option to view the **Current Text** attached to this widget. You can also use the **New Text** area to attach new text. For example, the button widget opens with the default text label, **A Button**, which you can then change to a text label that describes the action of this button.

Variables

All widgets that interact with a Perl script must have a variable name assigned to them, so that the Perl script can refer to this name. When you build templates, the widgets in the template can have a variable name. You can then change that name when you use this widget to create a new window.

Templates

The Templates window holds partial windows that you build using widgets. You can then invoke this set of Widgets in building a complete window. To create templates:

1. Build a window.
2. To save that window as a template, use either:

   File => ***Save*** => ***Template***, or

   File => Save as => Template

Scripts

The Scripting window holds completed scripts. After you build a window, if you save it as a script, it appears here.

File => ***Save*** => ***Script***

File => ***Save as*** => ***Script***

Steps in making scripts and windows for scripts.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To develop script, follow these basic steps:

1. Use the Scripttool to develop the part of the script that defines the interaction with Undertow:
   - Use hard-coded input.
   - Print the output in plain text.
   - Verify that the script works as expected.
2. Encapsulate the script in a subroutine, and save the subroutine.
3. To create the window (or GUI), use the **Builder** in Undertow.
   - Configure the **command** attribute of the **Apply** button, so it invokes the saved subroutine.
   - Verify that the output is still correct.
4. Remove the hard-coded input in the script.
   - Replace this input with data obtained from the GUI.
   - Verify that the results are as expected.
5. Send the output of the subroutine to the GUI, rather than printing it. Then verify correctness once again.

Examples of creating a simple script, and building windows.

To create a script, type it into the text area and save it. The following examples show you the steps to follow, to create some simple scripts.

Example #1: Creating A Simple Window

1. Open **Builder** from the scripts window.

   For example, **Window** => **Show Selector => Scripts** opens the script window**.**
2. When the script window opens, click on **Builder**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

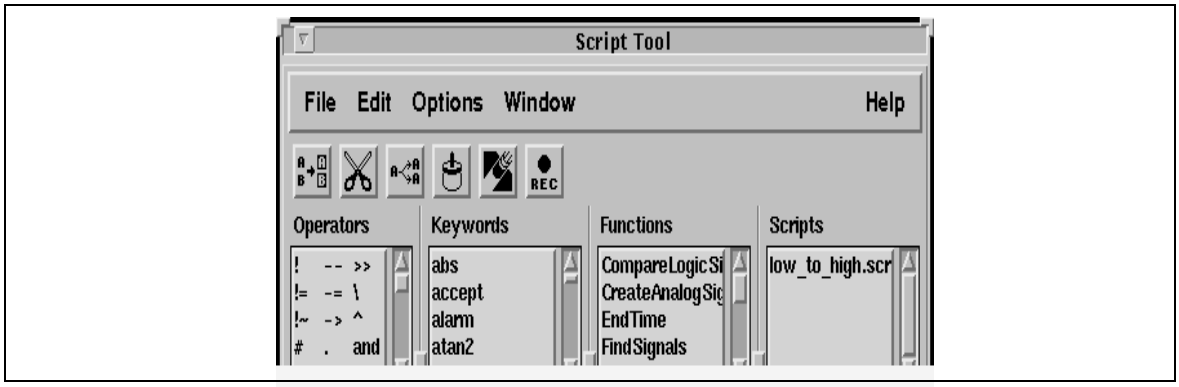


FIGURE 100. Script Tool Window

Creating a Label

1. Click on **Label** from the widgets list.

   The **Results** window opens, with a **Label** widget in the upper left corner. The label on the Label widget reads **A Label** by default.

2. Use the middle mouse button to drag the widget one centimeter down and one centimeter to the right.

3. Place the mouse over the new **Label** widget.

4. Using the right mouse button, click on the widget.

   A menu pops up.

5. Select **Text** from the popup menu.

   A dialog box titled **Text ?** pops up.

6. In the **Text ?** dialog, enter **Thing One:** under the words **New Text:**

7. In the **Text ?** dialog, click **Apply**, and then click **Close**.

8. Click on **Label** from the widgets list.

   The new **Label** widget appears in the upper left corner again.

9. Drag the new **Label** widget below the **Thing One:** label.

10. Change the text for this second label to **Thing Two**.

11. Click on **Entry** from the widgets list.

12. Drag the Entry widget to the right of the **Thing One:** label.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Setting a Variable

1. Move the mouse over the new **Entry** widget.
2. Using the right mouse button, click on the widget.

   A menu pops up.
3. Select **Variable** from the popup menu.

   A dialog box titled **Variable ?** pops up.
4. In the **Variable ?** dialog, enter **thingOne** under the words **New Variable:**
5. Click on **Entry** from the widgets list.
6. Drag the new **Entry** widget to the right of the **Thing Two:** label.
7. Change the widget variable for this second entry to **thingTwo**.
8. In the **Variable ?** dialog, click **Apply**, and then click **Close**.
9. Click on **Button** from the widgets list.
10. Drag the new button widget until it is beneath the *labels* and *entry* fields.
11. Change the text of the button to read **Swap**.

Adding New Perl Code

1. Press the right mouse button.

   A menu pops up.
2. Choose **Command** from the popup menu.

   A dialog titled **Command ?** pops up.
3. In the **Command ?** dialog, enter the following lines of Perl code, underneath the words **New Command**:

```
$thingThree = $thingOne;
$thingOne = $thingTwo;
$thingTwo = $thingThree;
```

4. In the **Command ?** dialog, click **Apply**, and then click **Close**.
5. Click the left mouse button in the *entry* field for thing one, and enter **1111**.
6. Click the left mouse button in the *entry* field for thing two, and enter **2222**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

7. Click the **Swap** button.

Saving Your Script

1. In the menus, select ***Options*** => ***Script type***.

   The **Script Type** dialog pops up.

2. Set the script type to **script**.

3. In the menus, select ***File*** => ***Save As***.

4. The file selection box opens.

5. In the file dialog, enter **swap** in the entry at the bottom.

6. Click **Accept**.

Creating a Template

Templates are reusable script fragments. For example the commonly appearing pair of buttons, **Apply** and **Close**, are a perfect application for a template.

1. To start creating a template, either:
   - Click the **Select All** button**,** and then click the **Cut** button (the left-most) on the Builder toolbar, to clear all current window buttons and start with an empty template in the current Builder window, or,
   - Restart **Builder**, to open a new Builder window with an empty template.

2. Click on **Button** from the widgets list.

   Builder creates a new button.

3. Place the mouse over the new button.

4. While pressing the shift key, use the middle mouse button to drag the right edge over of the new button to the center of the results window.

   The button resizes as you drag it.

   As you hold the right mouse button, a menu pops up.

5. From the popup menu, change the text of the new button to **Apply**.

6. Click on **Button** from the widgets list.

7. Drag the new button to the right of the **Apply** button.

8. Resize the new button to match the **Apply** button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

9. Change the name of the new button to **Close**.

10. Using the menu that the right mouse button pops up, change the command of the **Close** button to:

```
DialogQuit ($top);
```

11. Reduce the height of the **Results** window until it is only slightly taller than the **Apply** and **Close** buttons.

12. Center the two new buttons vertically in the window.

The template is complete, except for creating the command behind the **Apply** button. What actions occur when you click **Apply**, depends on what the script that uses this template will do. So leave this command empty for now.

Saving a Template

To save the template, follow these steps:

1. In the Builder pulldown menus, select ***Options => Script type***.

   The **Script Type** dialog pops up.

2. In the **Script Type** dialog, set the script type to **template**.

3. In the Builder pulldown menus, select ***File => Save As***.

   The file selection dialog box opens.

4. In the file selection box, enter **apply_close** at the bottom, and click **Accept**.

   Note: Do not use dots in the names of scripts or templates.

   When the script template has been saved, and is ready to be imported into a script, it appears in the templates list.

Example #2

1. Start the **Scripttool** from Undertow.

2. Click in the text area.

3. Enter the following text:

```
print "hello\n";
```

4. Click **Apply**.

   Your script prints hello to the results window in the script tool. Script tool always sends the output of the scripts it runs.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

5. Start **Builder** (which is itself a Perl script) from the Scripttools, as described below.

Starting Builder

1. If the **Scripts** list-box is not visible in the Scripttool, select ***Windows => Show Selector*** from the pulldown menus.
2. Click on **builder** in the **Scripts** list-box of the Scripttool.
3. Click **Apply**.

   The **Builder** window opens:
4. Click in the text area in **Builder**.
5. Enter the following line and press **Return**:

   ```
   print "hello\n";
   ```
6. Click the **Apply** button in the Builder.

   The output of your second script, again the word hello, again appears in the Scripttool results window. This occurs because Scripttool is running **Builder**, and Builder is running your second script. But unlike Scripttool, Builder does not create a results window for the output of the scripts it runs. Instead, it prints that output as if it were its own.

   Builder *does* include a results window, but uses that window to display the *GUI* output of the script it builds, rather than the *textual* output. The results window is the big, empty, square window that pops up when you click **Apply** in Builder. It is empty because, so far, your script in Builder does not create any graphical components; it contains only a **print** statement.

Adding A Label

1. In the **Templates** list-box, click on **Menubar**.

   Builder places the menu bar into the upper left corner.
2. In the **Templates** list-box, click on **Toolbar**.

   Builder places the toolbar directly underneath the menu bar.
3. In the **Widgets** list-box, click on **Label**.

   A label widget with the words **A Label** appears in the upper left corner of **Builder**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Note: ***Builder*** *does not automatically decide where to place any other objects that you create, nor does it know yet what text to place in the label.*

4. Using the middle mouse button, drag the label until it is directly underneath the toolbar, and then move it slightly to the right.

5. To configure the text on the label, press the right mouse button over the label, and select **Text** from the pop-up menu.

   A dialog opens, asking you to enter the new text that will appear.

6. Enter **template** in the entry field, underneath the words **New Text:**

7. Click **Apply**, and then click **Close**.

Adding List Boxes

1. In the **Widgets** list-box, click **Entry**.

   An entry widget appears in the upper left corner of Builder.

2. Drag the entry widget until it is immediately to the right of the new label.

3. In the **Templates** list-box, click on **Threelist-boxes**.

   Builder places a frame containing three list-boxes, without scrollbars, in the upper left corner of the screen.

4. Turn highlighting on before attempting to drag the frame.

5. In the pulldown menus, select ***Debug*** **=>** ***Highlight***.

6. Slide the mouse across the GUI that you are creating.

   Builder shows the widgets in reverse video while the mouse is over them, so you can see what you are about to drag.

7. Drag the frame containing the list-boxes straight down, until it is beneath the label and entry field.

   Note: Leave room for one more row of labels between the label and entry field, and the list-boxes.

Labeling the List Boxes

1. In the **Widgets** list-box in Builder, click three times on **Label**.

   Builder adds three labels, one on top of another, in the upper left corner of the **results** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Drag each label to a point just above each of the three waistbands.

3. Slide the mouse over the first (left-most) label.

4. Press the right mouse button, and select **Configure** from the popup menu.

   The configuration dialog opens. Two list-boxes appear:

   - The left box is filled with all of the existing widgets in the **Builder** program.
   - The right box is filled with all of the configurable attributes of the widget that you are configuring.

5. Scroll to the bottom of this list, and click on **text**.

   The words **A Label** appear in the **Current Value** text widget.

6. Click in the **New Value:** text area, enter **Widgets**, and click **Apply**.

7. Slide the mouse button over the second (middle) of the three labels that you created.

8. Refresh the configuration dialog the same way you opened it the first time, using the right-mouse-button popup menu.

9. Scroll down the right list-box, choose **text**, and change the text of this label to **templates**. Also change the text in the right-most label to **Scripts**.

10. Click **ApplyClose** in the templates list.

Creating and Activating the Window

1. Drag the frame containing the **Apply** and **Close** buttons, until it is just below the text area, and center it in the window.

   Note: This step is difficult, because there is not much room to grab the frame.

2. Resize the window, until the height fits the graphical elements in the window.

   *Note: You are creating a duplicate of the **Builder** window!*

3. Attach the **Apply** button to a script that does what you want.

4. Attach the other widgets of your new GUI to provide input to that script, and also to display the results.

Example #3: Script to measure analog signals

The following is an example of creating a script, and using the Veritools **Builder** to build a window that you can use to execute this script.

The following script, named measure.script, was designed to do the following:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Read the names of two analog signals selected on the **Waveform** display.
- Read the values of **T0** and **T1**, the time cursors that define the outer bound, measuring between two thresholds on the two signals.
- Place the **T1** cursor on a user-defined threshold of the first signal.
- Place the **T0** cursor on a user-defined threshold of the second signal.
- Measure the difference between **T0** and **T1**, and place the result in the results output location.

```
$cursor1time = GetCursor (0);
$start = $cursor1time;
$start2 = $cursor1time;
print "cursor1time = $cursor1time \n";


$cursor2time = GetCursor (1);
$end = $cursor2time;
$end2 = $cursor2time;
print "cursor2time = $cursor2time \n";


&PlaceCursor(1, $cusror1time);
($signal, $signal2) = &GetSelectedSignals();
print "signal = $signal, signal2 = $signal2 \n";
if (not &GoToValue ($signal2, $threshold2)) {
  print "No value found on $signal2 \n";
}
$time2 = &GetCursor (0);
print "time2 = $time2 \n";


&PlaceCursor (0, $cursor1time);
print "just placed cursor 0 at $cursor1time \n";
if (not &GoToValue ($signal, $threshold)) {
  print "No value found on $signal \n";
}
$time1 = &GetCursor (0);
print "cursor 0 set on $signal set at $threshold at time= $time1\n";
&PlaceCursor (1, $time2);
print "cursor 1 set on $signal2 set at $threshold2 at time= $time2\n";
$measure = &GetCursor(1) - &GetCursor(0);
print "measure: " . $measure . "\n";
$results = $measure;
```

You can now build a window for this script, that will do the following:

- Hold the names of the two selected signals in two text areas, called **Signal**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Provide two threshold text areas, where you can enter the values of the thresholds:
  - The upper threshold area is for the first signal.
  - The bottom threshold area is for the second signal.
- Provide a **Start** and **End** text area for the boundary of this measurement, as defined by the locations of **T0** and **T1**.
- Provide a result area for the **T0** to **T1** measurements.
- Provide **Apply** and **Close** buttons, to control the operation of this window.

Creating Templates

You can use the **Builder** to build a window. In the Script tool:

1. In the pulldown menus, select ***Window => Show Selection => Scripts***.
2. In the Scripts list, click on **Builder**.
3. Click **Apply** at the bottom of the Script Tool.

   The Builder window opens.

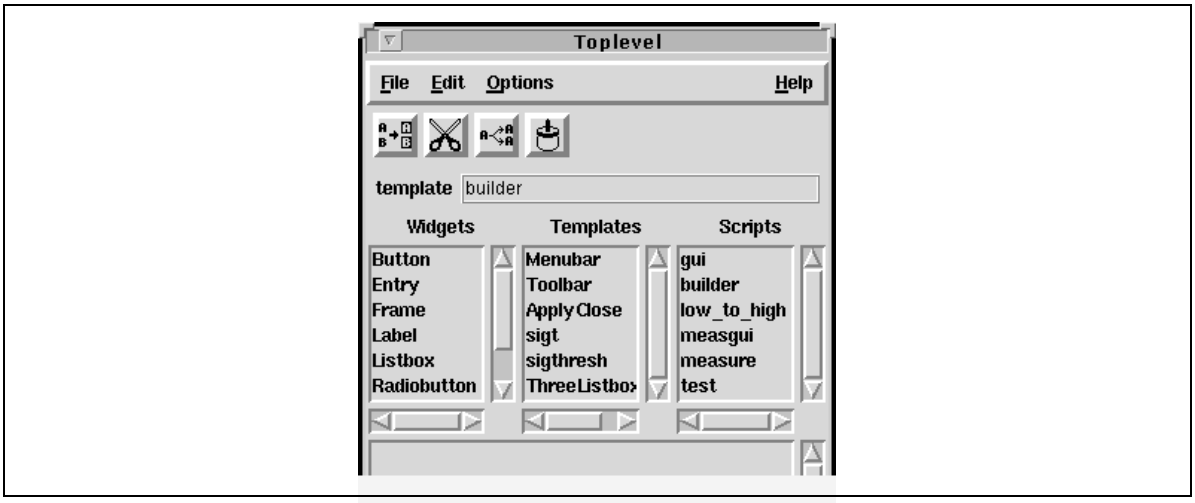


FIGURE 101. Script Tool

4. In the pulldown menus of the **Builder** window, select ***Templates => Menubar***.

   A menu bar opens in a window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

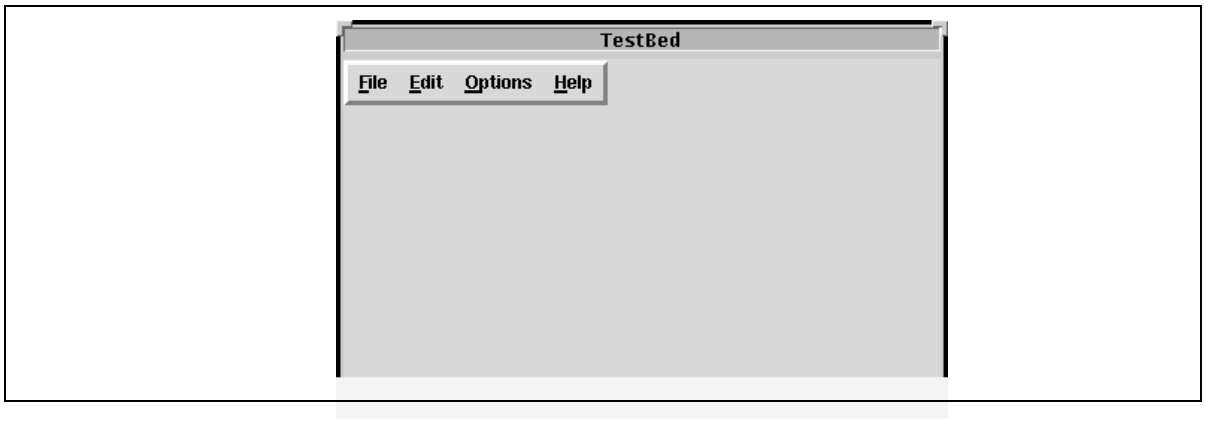


FIGURE 102. Menubar in Builder Window

5. In the pulldown menus of the **Builder** window, select ***Templates*** => ***Toolbar***.

   A Toolbar appears below the window in the frame.

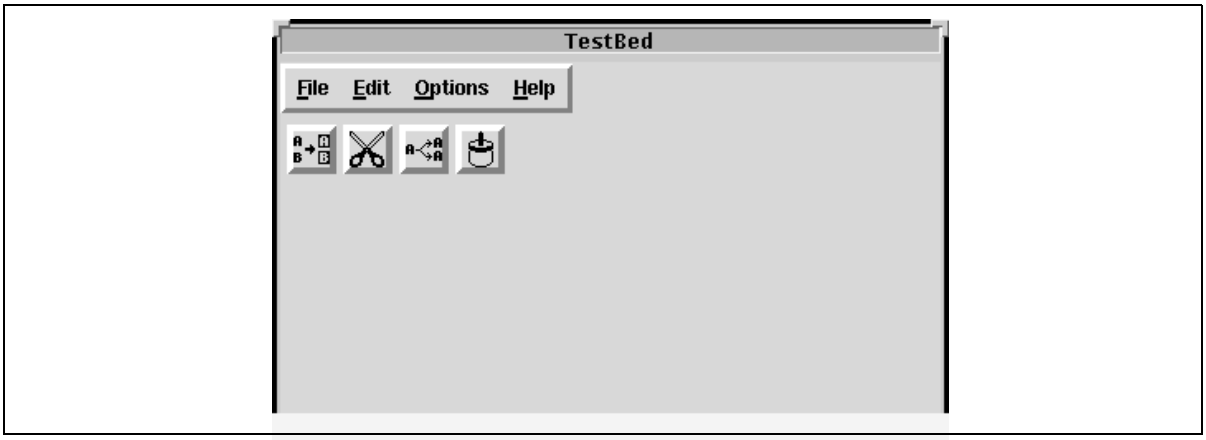


FIGURE 103. Menubar and Toolbar in Builder Window

6. In the pulldown menus of the **Builder** window, select ***Templates*** => ***sigt***.

   A template opens, with entries for signal, threshold, and start and end times to bracket the threshold.

7. Move this template down, to position it properly beneath the toolbar.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

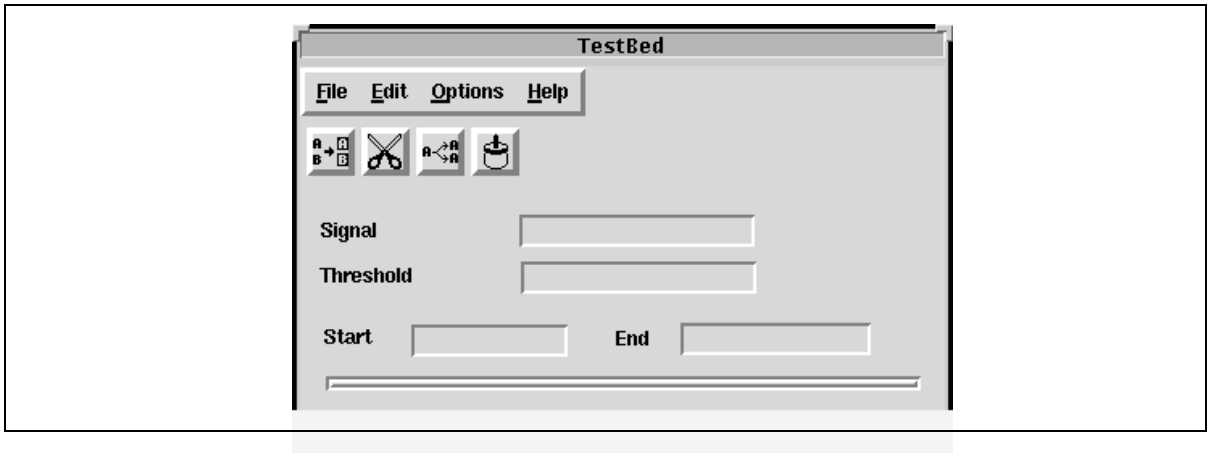


FIGURE 104. Template in Builder Window

8. In the pulldown menus of the **Builder** window, select ***Templates*** **=>** ***sigthresh***.

   Another template opens, similar to **sigt**. However, this template also contains an **Update** button, which you can use to move the **T0** and **T1** cursors. You can then bracket the threshold again on signal 2.

9. Drag this new template down, until it is just beneath the **sigt** template.

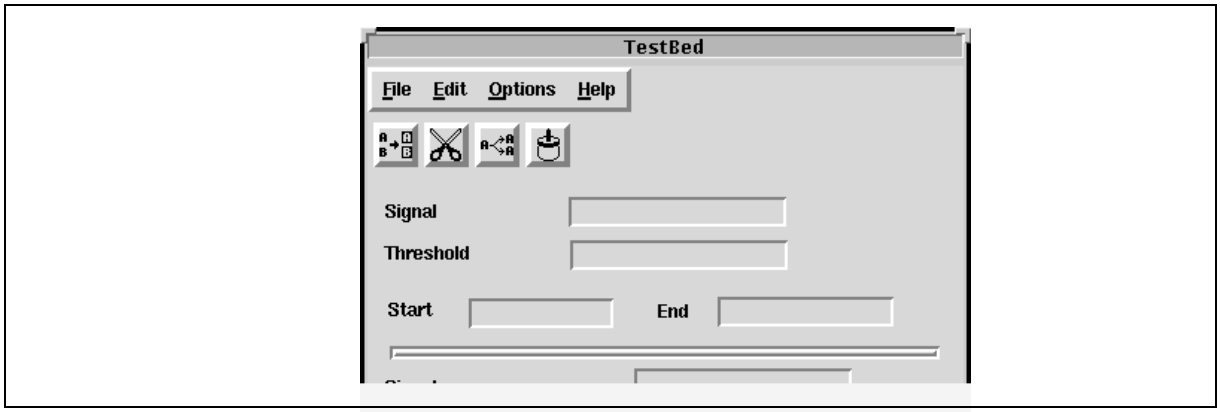


FIGURE 105. Second Template in Builder Window

Adding Widgets

1. In the widgets list, click **Entry**.
2. Reposition the entry to just below the templates.
3. Place the mouse cursor over this entry widget.
4. Use the right mouse button to open the popup menu, and select **Variable ?**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The dialog titled **Variable ?** opens.

5. When the dialog opens, enter **result** in the entry field for **New Variable:**
6. Click **Apply**, then click **Close**.

Adding Labels

1. In the widgets list. click on **Label**.
2. Drag the new label to the left of the entry widget that you just added.
3. Using the popup menu that appears when you press the right mouse button, select **Text**.
4. Change the text of the new label to read **Result:**
5. In the templates list, click on **ApplyClose**.

   A template containing **Apply** and **Close** buttons appears in the upper left corner.
6. Drag the template down to the bottom of the window.

   Whenever you use a template in a script, Builder places the contents of the template in a frame (in this example, the frame contains both the **Apply** and **Close** buttons).

   *Note: You must drag the entire frame, not just the **Apply** button or just the **Close** button. To do this, place the mouse cursor in the space between the two buttons, before you drag the frame.*

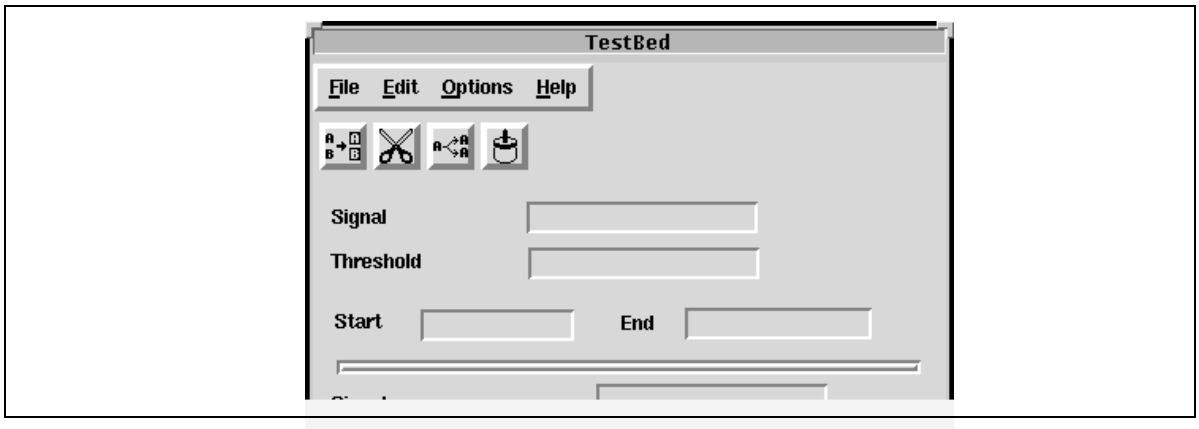


FIGURE 106. Completed Template in Builder Window

Attaching Scripts to Buttons in the Template

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1. To set the command behind the **Apply** button, again press the right mouse button on the **Apply** button.
2. When the dialog titled **Command ?** opens, enter the following in the area labeled **New Command:**

   &DoScript(“measure”);

   Builder attaches the measures.script script to the **Apply** button. That is, when you click **Apply**, the measures script executes.
3. In the command window, click **Apply**, and then click **Close**.

   Your script is ready to save.

Saving Your Script

1. From the pulldown menus, select ***Options => Script Type***.
2. Set the script type to **script**.
3. From the pulldown menus, select ***File => Save As***.
4. To save the script under a new name, select the script name from the **scripts** list.

   In this example, click on the name **measuregui**.
5. Click **Apply** again.

   When you click **Apply**, the **measuregui** script runs, and invokes the **measures** script.
6. Read the **measures** script, to understand what it is doing.
7. Place your **T0** and **T1** cursors.
8. Enter the appropriate values in the **threshold** entry fields.
9. Click **Apply**, to view the results of the **measure** script.

Plotting Functionality

This section describes how to create plots from Undertow data.

Continuously plot waveforms as simulation progresses

The closest thing Undertow has to *marching* waveforms, is a fast-reread operation. Although it must reread the entire file on reload, it reloads the file

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

very quickly.

Use ***File =>Reread Dumpfiles***.

Independent, user-selectable offsets, for each axis of each signal

Undertow meets the requirement for independent, user-selectable offsets, for each axis of each signal, on a multi-signal plot, but only for analog waveforms.

1. Select the signal of interest.
2. To change offsets for a signal, select the ***Tools => Analog Analyzer => Selected***
3. ***Properties*** menu item.

Support for multiple x axes

You can assign *one* axis to each plotted waveform. To assign an axis, select one of the following:

- ***Window => Add New Pane***, or
- ***Arbitrary X***, or
- ***Window => Add New Pane, Time***.

If the selected signal has a time-based **x** axis, then select ***Window => Add New Pane, Time***. Otherwise, select ***Window => Add New Pane, Arbitrary X***.

Support for multiple y axes

You can assign *one* axis to each plotted waveform.

Undertow lets you use two Y axes (left and right). You can use either axis for any specified signal, but there are *only* two axes. You cannot overlay 3 signals with independent axes.

1. Select the signal of interest.
2. To select whether to use the left or the right axis for the selected signal, select the ***Tools => Analog Analyzer => Selected Analog Waveform Properties*** menu item.

User-selectable linear or log axes

1. Select the signal of interest.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select ***Tools*** => ***Analog Analyzer*** => ***Selected Analog Waveform Properties***.

Show Verilog code, related to a signal in another window

You can use Undertow Suite to view the exact Verilog source code associated with a selected signal transition.

Generate eye diagrams

1. Select the ***Tools*** => ***Analog Analyzer*** => ***Eye Diagram Analyzer*** => ***New*** menu item.
2. To specify signals from multiple files, use the format:

   /1/*netname*

   where **1** is the number corresponding to the dumpfile.

View grouped signals

You can view grouped signals as hexadecimal, octal, decimal, or binary values, or as user-definable mnemonics.

1. Select the ***Options*** => ***Bundle Signals*** menu item.
2. Select the appropriate ***Format*** => ... menu item.

Plot the actual data points

Use the ***Format*** => ***Analog, Points*** menu item.

Graphics Capabilities

This section summarizes the graphics capabilities in Undertow.

Choose a color and pattern, to display each signal or group

Undertow does not allow patterns. However, you can use the ***Format*** => ***Analog Color*** menu item to choose the color to use for each signal.

To be able to choose from more than 27 colors, you can declare more colors in your .Xdefaults file. To do this:

1. Attach the following lines into your .Xdefaults file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
UNdertow.analogColors: white red green blue cyan yellow \
SpringGreen4 LightPink LimeGreen \
MediumSpringGreen CadetBlue \
LightPink1 \
MediumPurple1 LemonChiffon1 \
aquamarine1 orange \
LightSeaGreen HotPink khaki1 \
SpringGreen2 PaleGreen4 LightYellow2 \
green4 gold3 \
LightSkyBlue LightCyan \
LightCoral MintCream
```

2. Add colors by means of the same syntax.
3. To run **xrdb**, type;

   ```
   xrdb ~/.Xdefaults
   ```

4. Restart Undertow.

User-selectable title, axes labels, fonts, sizes, styles

Each pane can have a title.

- Select the ***Windows*** => ***Tags*** => ***Title Tag*** menu item for Titles.
- Select the ***Format*** => ***Properties*** => ***Fonts*** menu item for Fonts.

User-selectable grids, tics, value labeling of axes

Select the ***Options*** => ***Properties*** => ***Grids and Scales*** menu item.

Add text and arrows, at arbitrary orientations, to graphs

- Select the ***Window*** => ***Tags*** menu item, or
- Right click on the tag, and select **Configuration**.

Modify the size and aspect ratio of a plot

Select the ***Window*** => ***Waveform Heights*** menu item.

User Interface Functionality

This section summarizes functionality in the Graphical User Interface (GUI).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Support user-configurable bindkeys

The Undertow application default file specifies the current value of the bind keys. This file is located in the ut installation directory. The list in the Undertow file is under:

UNdertow_Displayer*display_WaveWindow.translations

which shows the predefined Bind Keys.

For example, to set a horizontal zoom into **Ctrl z**, which is the default:

1. Put the following lines into your **~/.Xdefaults** file:

   UNdertow_Displayer*display_time_clip.translations: #override\n\

   Ctrl<Key>z: commandZoomIn()\n\

2. After you set any new defaults, run the following command:

   xrdb ~/.Xdefaults

3. To read the changes, restart Undertow.

Synchronize inspection operations between plots in separate windows

1. Select the ***File*** => ***Start New Undertow*** menu item.
2. Select either the ***File*** => ***Sync*** or the ***Window*** => ***Sync Panes*** menu item.

Synchronize zoom and pan operations on plots in separate windows

Select the ***Window*** => ***Sync Panes*** menu item.

Signal selection and grouping

Use the **Groups** button.

Search for signals by matching regular expressions

The signal chooser supports wildcard matching, but it does not support full regular expressions. This capability meets the needs of most Undertow users.

Save the program state in the state file

Select the ***File*** => ***Save*** menu item.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Initialize the program to its previous state, from the state-file

Select the ***File*** **=>** ***Open/Close File*** menu item.

Note: To achieve the same result, you would have to add each individual dumpfile, in the same order, before upload the state-file. The ***save*** *file does not include information about what dumpfiles you loaded.*

A plot can fill an entire window

1. Select the plot.
2. Click the **Fit to Pane** button.

Support options to create and edit waveforms.

You can save waveforms, and use them as simulation stimulus waveforms.

For example, you can select a waveform to output as a piece-wise linear waveform, and use it as an input for *hspice*.

1. Select a signal.
2. Select the ***Options*****=>** ***Get Information*** **=>** ***Dump Signals*** menu item.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. If needed, edit the waveform so you can use it as an input stimulus waveform.

Edit a waveform expression, without reentering the entire expression

1. Select a waveform.
2. Select the ***Tools*** => ***Analog Analyzer*** => ***Analog Functions, Edit Selected*** menu item.

Analog waveform processing capabilities

This section summarizes the processing capabilities for analog waveforms, that are included in Undertow.

Transform y axis of signal using arbitrary combinations of functions

Undertow uses Perl to meet this requirement. The **Analog Functions** menu, and the **plot X vs. Y** capabilities, also provide a lot of flexibility.

Transform x axis of signal using arbitrary combinations of functions

Undertow uses Perl to meet this requirement. The **Analog Functions** menu, and the **plot X vs. Y** capabilities, also provide a lot of flexibility.

Perform functions on signal data

Undertow supports the basic math operators, plus several functions, including the ones listed. Select the ***Tools*** => ***Analog Analyzer*** => ***Analog Functions*** menu item.

It is unclear how good this function's complex math is, and it might not accept some complex constants.

Compute 10-90% and 20-80% rise and fall times of signals

Select the ***Tools*** => ***Analog Analyzer*** => ***Rise Time Analyzer*** and ***Fall Time Analyzer*** menu items.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Access data from an individual single sweep, after a parametric sweep

Circuit simulators, such as *Spice*, allow two-dimensional sweeps. You might then need to perform intersweep comparisons. For example, if the value of a resistor is swept among 10 sweeps, you might need to compare the ten sweeps to a *reference* sweep, such as the first sweep.

An example of the syntax is:

/1/I(vin1(real){0 (}), /1/I(vin1(real){1 (}), /1/I(vin1(real){2 (})

FFT

Undertow includes an **FFT Analog Function**. For an example of how to use it, see Section , “Analog Tools”.

Allow access to “Inoise” and “Onoise” noise data in AC analysis

The node names are **innoise**, **inoise(mag**, **outnoise**, and **onoise(mag**, respectively.

AC or Fourier results should be viewable by magnitude

An Analog Function, named **MAGN**, returns the magnitude of a complex number. You can use this function to find magnitudes for any complex signal

FFT

FFT Interface

Undertow allow user to perform DFT/FFT(Discrete Fourier Transform, Fast Fourier Transform) with a simple, easy to use, menu. To invoke FFT, do the following:

Invoke FFT either click on the graph icon on the icon menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

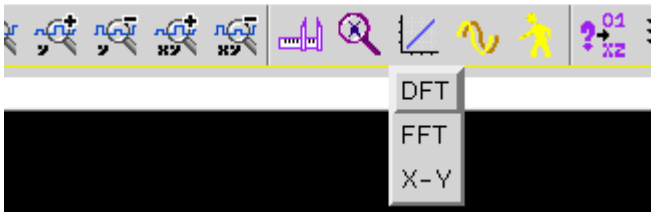


FIGURE 107. Invoking FFT function from the icon menu

FFT Window Menu

FFT function window consist of the following Functions:

TABLE 46. FFT Window Functions

Functions	Explanations
Output Format	This Option control the output of the FFT, It will plot against Magnitude/Phase as user specified.
Transform type	User can choose between FFT/DFT transform. Note that for FFT, the number of sample should be in 2^n form.
Signal Name	The signal that FFT/DFT will performed.
Sample frequency/No. of Samples	Sample frequecny for sampling or number of sample user wish to have in the period specifed by Start and End Time
Start x:	Starting time for FFT
End X:	Ending Time for FFT
Result: Index	X-axis, the axis will be in Index if this is specifed
Results: Frequecy	The X-axis will be in frequency if this is specified
Results: Period	The X-axis will be in Period if this is specified

Note: the number of sample for FFT has to be in the power of 2, For example, 256, 512, 2048, 16384, 32768, 65536, 131072, ... and so forth

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

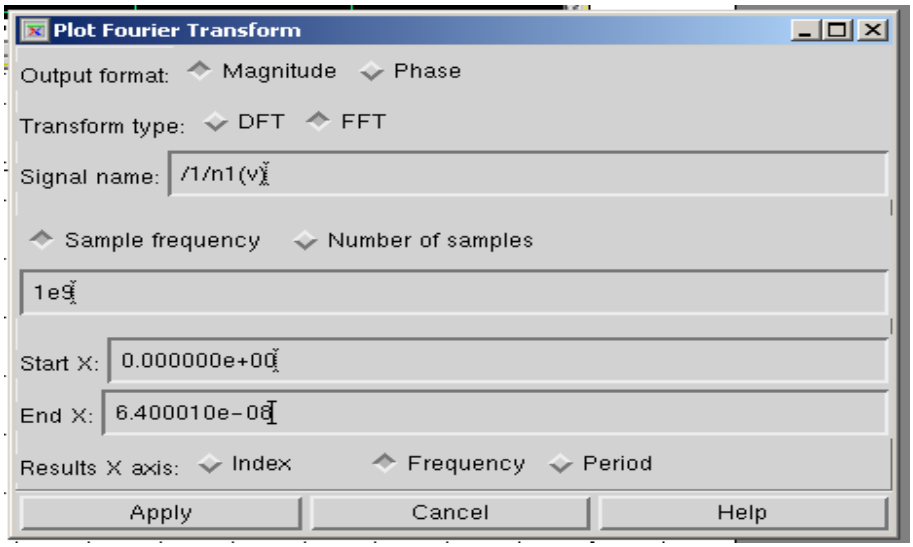


FIGURE 108. FFT/DFT Window

Note: If using Number of Sample as input for the FFT/DFT, specify the Start Time and End Time accordingly as the results can be greatly affected by the number of samples being sampled in the specified period.

After FFT/DFT is performed, waveform window will show a continuous curve. If users wish to view the discrete point for each harmonics, they can do **Format->Analog Bar Graph** to view the graph in the discrete Bar form. Use Option->Get Informations->Dump Signal to dump out a

Retrieving the Data Point

After FFT/DFT is performed, user can dump out the data in text form to view the results of the FFT/DFT. **Options->Get Informations->Dump Signals/Dump Signals, Tabular** to dump out the signal in text format. Users can save using **File->Save/Save As** to save the informations of copy and past to another file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

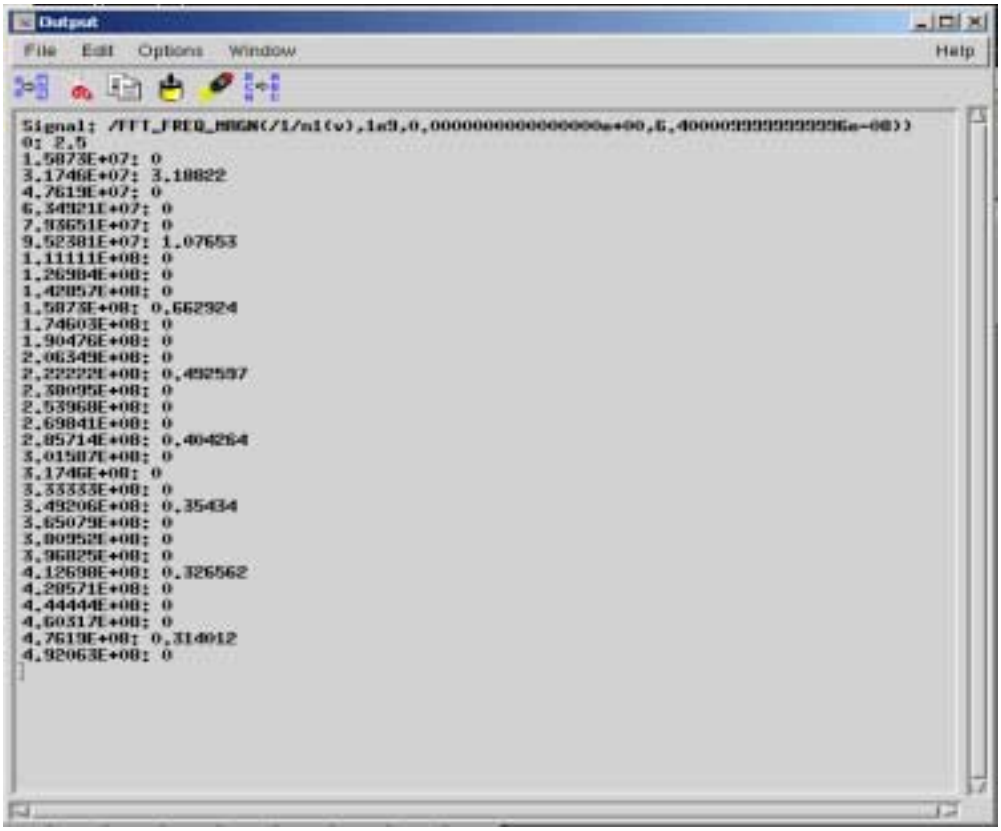


FIGURE 109. Dump signal Text output

Saving Signals

Action: File->Save

This command saves the current FFT signals and the output signals in a file. To Save the signals, **File->Save**, type in the save file name and click on **Save All (Environents),** the default extension for the save file is *".save"*

Restore Signals

Action: File->RestoreThis command restores the signals from an FFT file that you saved previously by Save Command. The Restore Sgnals command works the same as the Open File. To retrieve the wave form, Do File->Restore, click on the file name(default file filter is *".save"*). The signal names will appearSsed in the chooser windows and user can simply drag and drop the names into the waveform window to view the waveform.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Example

Here's a simple unit square-wave(height 5 with 32ns period). Click on the Plot Icon and select FFT

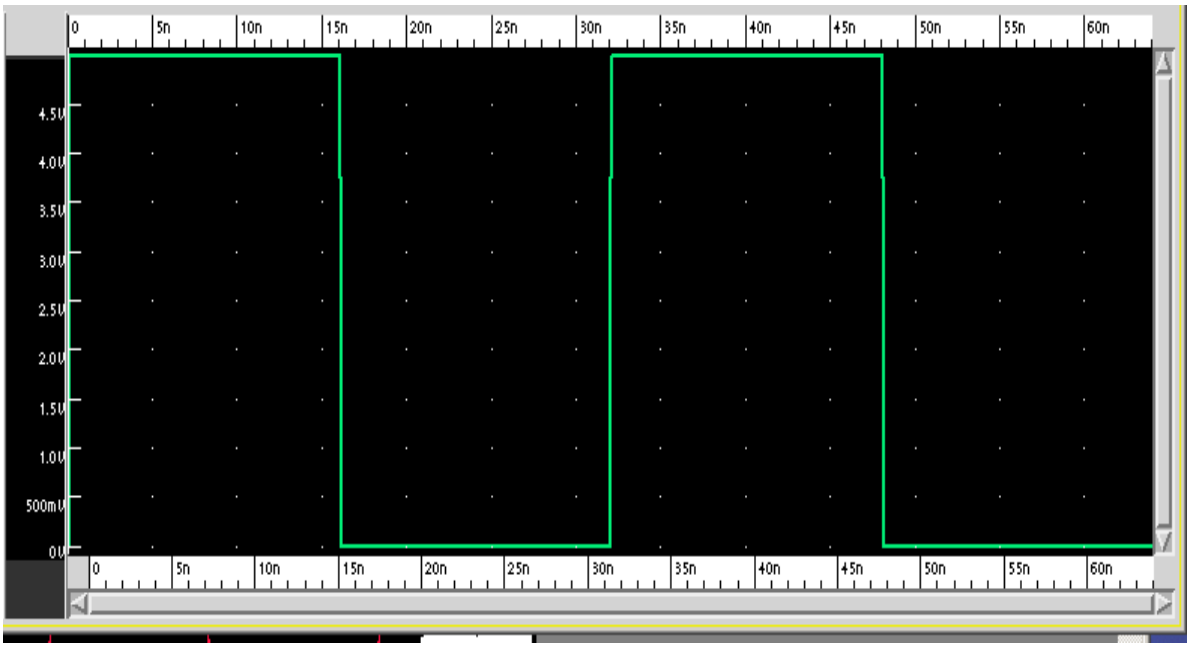


FIGURE 110. Square Waveform to perform FFT

For this waveform, we will sample it with 1e9Hz frequency and plot it against Magnitude vs Frequency.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

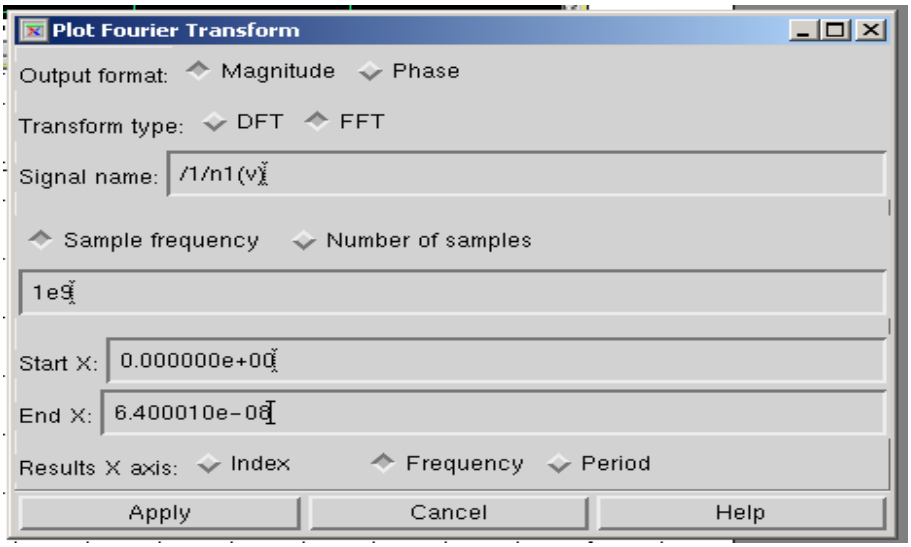
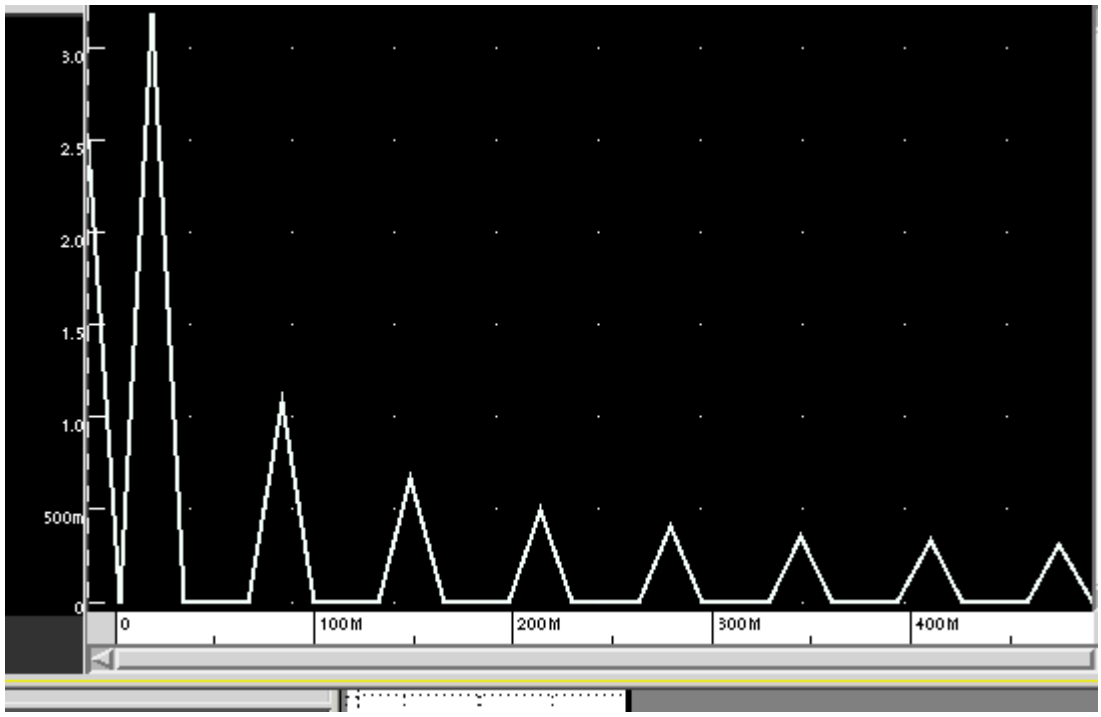


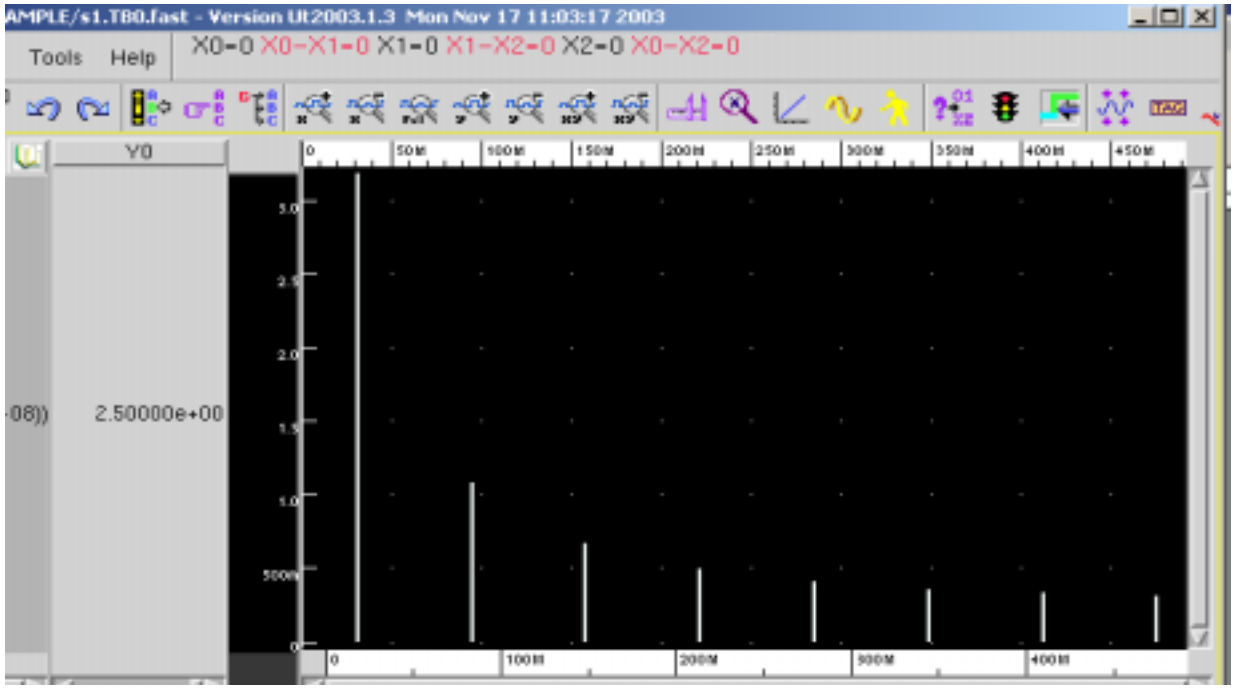
FIGURE 111. FFT Window

The resulting waveform is as following:



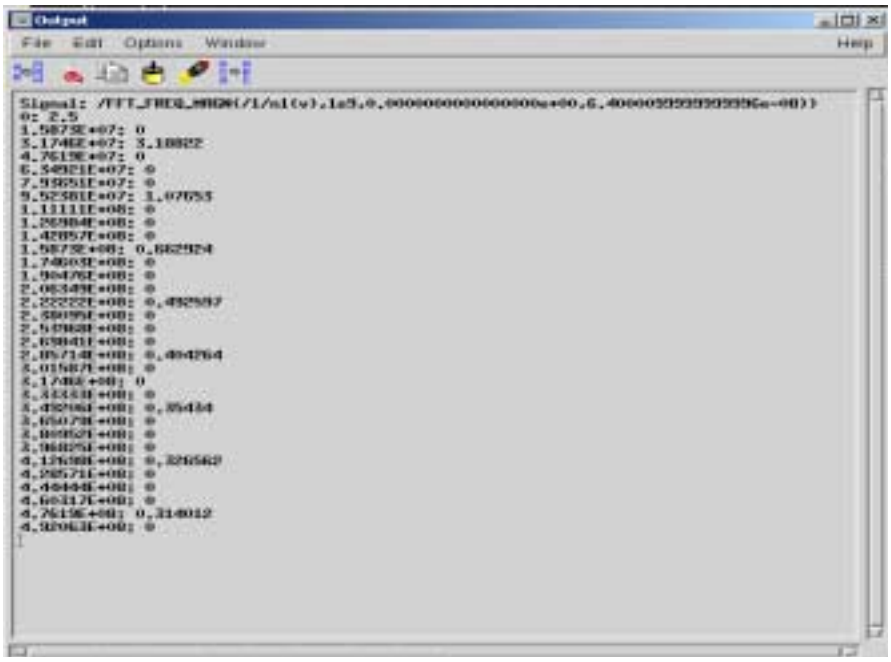
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To Change it to Bar Graph, go to **Format ->Analog, Bar Graph**



To Dump out the signal in text format:

Options ->Get Information -> Dump Signal to dump out all the signal point and its correspoinding frequency.



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Simulation results comparison and analysis

This section summarizes how to compare and analyze simulation results in Undertow.

Compare transition times, from digital simulation result files

Select the ***Tools*** => ***Test Analyzer*** menu item.

Compare signal levels of digital simulation result files

Select the ***Tools*** => ***Test Analyzer*** menu item.

Comparing results of simulations

Comparing simulation results can produce an ASCII report file, as well as a graphical display.

This is an option in the **Test Analyzer**.

Measuring time differences between signal transitions, on different signals

1. Select the signals.
2. Click the **Measure Tag** button.

Compare analog waveforms

You can use the ***Tools*** => ***Analog Analyzer*** => ***Analog Functions*** menu for many processing operations. For example, you can subtract two waveforms to compare them. If necessary, use the **clip** function in Perl, to specify the range of comparisons.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Absolute Value Function

A crude way to obtain an absolute value function is to square a value and then use its square root.

Compare crossing points over a user-selectable threshold

Undertow uses *measures* and *tags* to partly meet this requirement.

Compare analog waveforms to digital waveforms

You can convert an analog waveform to a digital waveform, and then compare the two digital waveforms. If necessary, use the **clip** function in Perl, to specify the range of comparisons.

Convert analog waveform to digital waveform

Select the ***Tools*** => ***Analog Analyzer*** => ***Convert to Digital*** menu item.

Search for a point of interest on a curve so that it can be marked

You can use ***Window*** => ***Navigator*** => ***Go To*** to search for some values. Specific Perl scripts perform peak and trough searches, and search for the *n*th crossing of a threshold.

Output file compatibility

This section summarizes issue regarding compatibility of output files generated in Undertow.

Encapsulated PostScript (EPS) graph output, color or black-

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

and-white

This requirement is met in Undertow 8.2.7.1.

1. Select the ***File*** => ***Print Signals*** menu item.
2. Select **print to file**.
3. Save the file with a .eps extension.

Note: You cannot use environment variables or tilde (~) in the pathname, but you can use relative pathnames.

PostScript (PS) file output, color or black-and-white

1. Select the ***File*** => ***Print Signals*** menu item.
2. Select print to file.

Note: You cannot use environment variables or tilde (~) in the pathname, but you can use relative pathnames.

x-y value pairs

The output file can be x-y value pairs for plotted signals, or specified signals and expressions, over selected range.

1. Select the ***Options*** => ***Get Information*** => ***Dump Signals*** menu item.

Save results of arbitrary expressions, for reload and comparison

You can save the results of arbitrary expressions, in a format that you can load again, to compare with other signals.

Select the ***File*** => ***Save*** and ***File*** => ***Restore*** menu items.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 6

Undertow Suite User's Guide

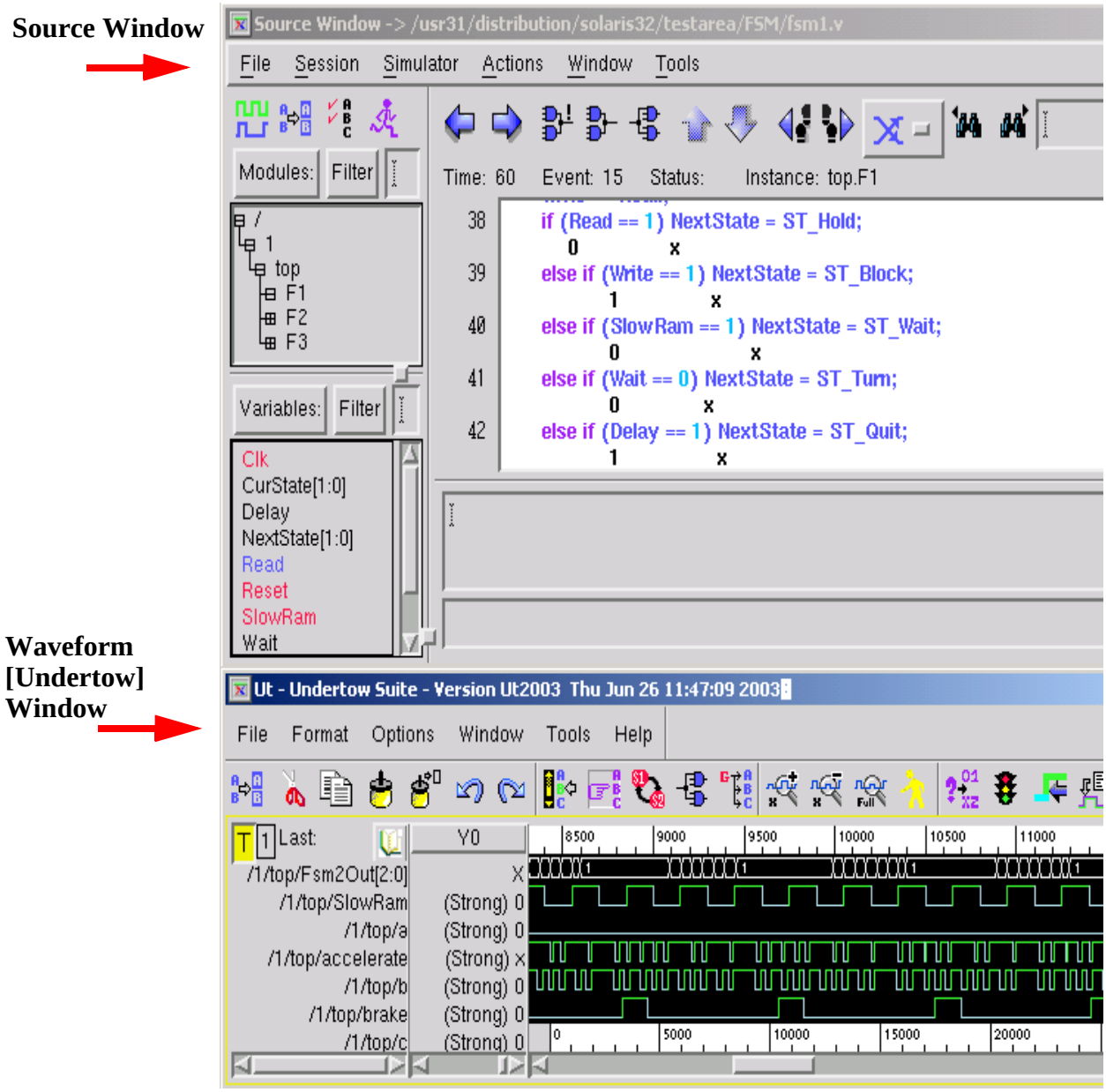
Undertow Suite Overview

This chapter describes the basic operations of the Undertow Suite. You can use this set of tools to debug source code for Verilog, VHDL, Verilog/VHDL mixed mode, and SystemC. This section introduces the main windows, describes how to start the Undertow Suite, and presents examples on how to use the main windows.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

When the tool is invoked ,it displays the windows as shown in Figure 112.

FIGURE 112. Opening Undertow Suite



Defining Mouse Controls

For using the New Undertow mode for the mouse controls, click Window =>Configuration => Mouse

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The following are the definitions of the mouse operations which will apply in the New Undertow Mode :

- **Select or click** - use *left* mouse button.
- **Double-click** - use the *left* mouse button to rapidly click twice.
- **Right-click** - use the *right* mouse button.
- **Drag and drop** - press the *middle* mouse button over an item and while still pressing down drag it to the new location
- **Left mouse button** sets the T0 cursor in the Waveform Window.
- **Middle mouse button** sets the T1 cursors and all additional cursors with an unbroken sequence of connected delta tags.

For example, clicking the left, middle, middle mouse sets the T0, T1 and another T1 cursor with a delta tag between the T0 cursor and the first T1 cursor and another delta tag between the first T1 cursor and the second T1 cursor.

- **Box zoom** - press the *right* mouse button and slide it over the area to be enlarged, release when box includes the desired area.

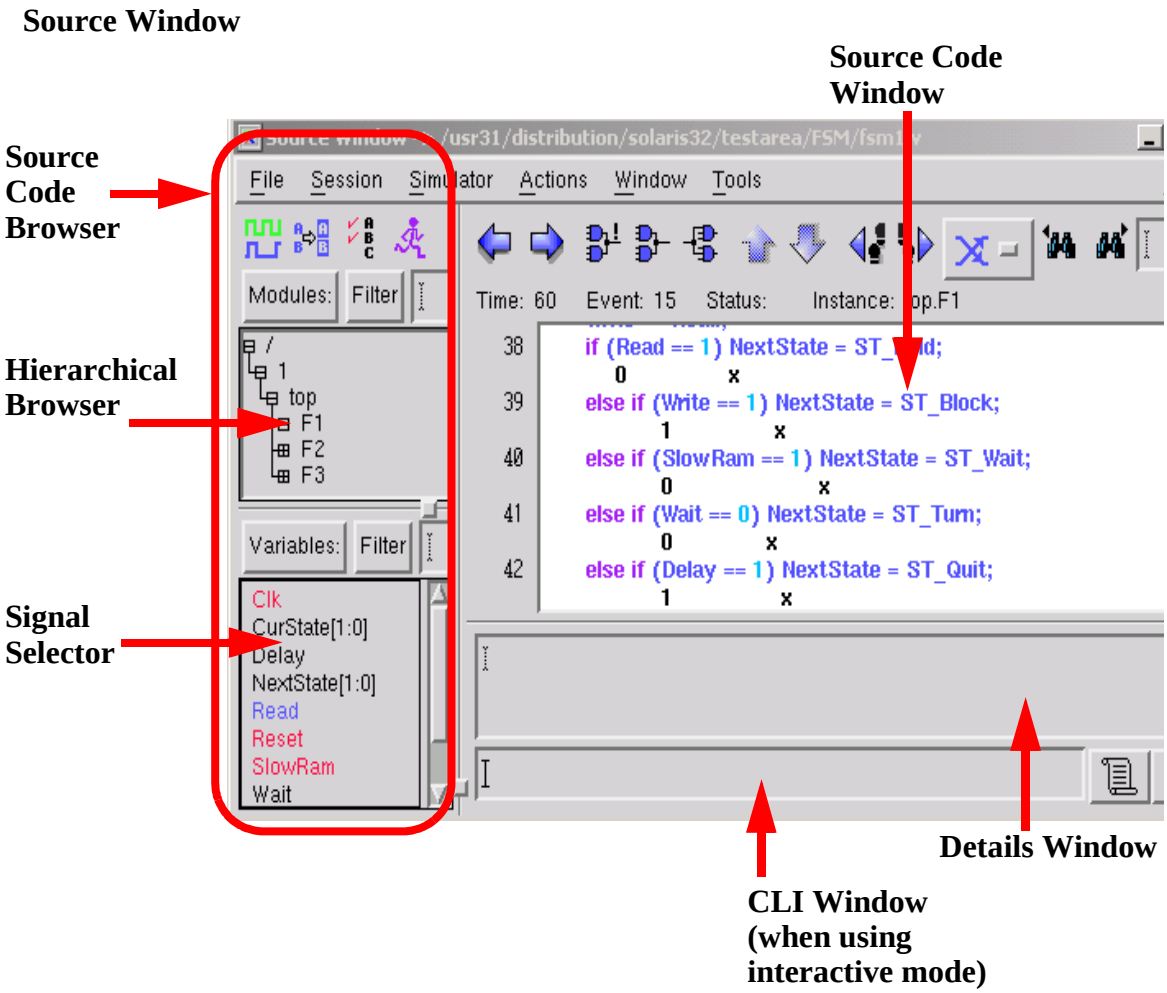
Source Window

The Source Window contains the Source Code Browser and Source Code Window at start up. The Details Window and CLI Window can then be activated as needed.

The Details window and the Command Line Interface, CLI, window, will also come up within the Source Code window when these are needed for some source code window operations. The Details window is used when ever multiple assignments or multiple loads are requested to be shown in source code window, or the Undertow Suite is used to run a simulator interactively. The CLI window is used only when the user has theUndertow Suite connected and running a simulator interactively. The CLI window allows the user to type in simulator commands, although the command drop down icon at the right side of the CLI window is the preferred method for selecting commands to be sent to an actively running simulation session.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 113. Reference example for viewing the Source Windows

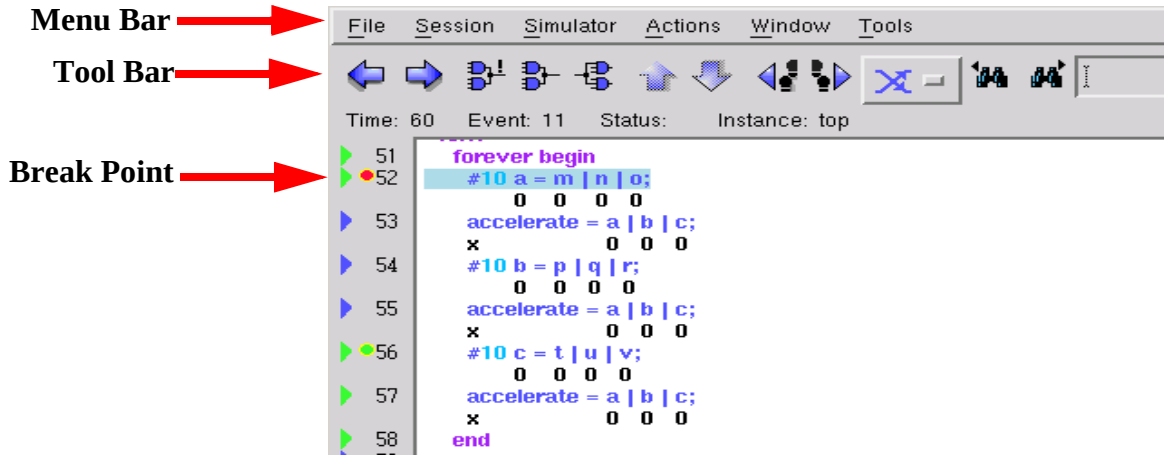


 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Source Code Window

Use the Source Code Window to view your source code (See Figure 114). You can also use this window to interactively control the simulator, if the simulator is currently running.

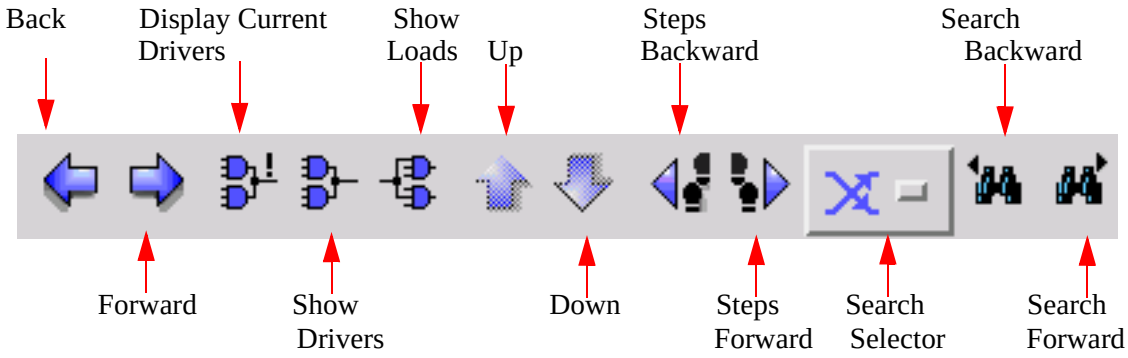
FIGURE 114. Source Code Window


Menu Bar
Tool Bar
Break Point
File Session Simulator Actions Window Tools
Time: 60 Event: 11 Status: Instance: top
51 forever begin
52 #10 a = m | n | o;
0 0 0 0
53 accelerate = a | b | c;
x 0 0 0
54 #10 b = p | q | r;
0 0 0 0
55 accelerate = a | b | c;
x 0 0 0
56 #10 c = t | u | v;
0 0 0 0
57 accelerate = a | b | c;
x 0 0 0
58 end

Controls on the tool bar let you:

- Step the simulator forward, in interactive mode.
- Step the simulator backward (if a trace file had been created), in either interactive or batch mode.
- Using these controls, you can step forward or backward without using an active simulator license.

FIGURE 115. Source Code Window tool bar



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The Source Code Window tool bar includes (See Figure 115):

- **Back/Forward** — Goes to the previous or next load/driver.
- **Display Current Drivers** — In a pop-up window, it displays the drivers for a selected signal for the specified time during the simulation.
- **Show Drivers/Show Loads** — Shows the drivers and loads, in the Source Code Window. Use the Back or Forward icon to step through the source code.
- **Up/Down** — If more than one driver or load are present, the Up and Down icons will be enabled. The user can see this by the fact that the arrows change from grey to black.
- **Steps Backward/Steps Forward** — Use these controls to step forward or backward by one time unit. This operation requires a trace file.
- **Search Selector** — Choose whether to search for a Transition, a Leading (rising) Edge, a Trailing (falling) Edge, a Value, or a String.
- **Search Backward/Search Forward** — Search forward or backward for the item chosen in the Search Selector.
- **Search String** — Use this text box in conjunction with the Search Selector items value and string. Type in the value or text string to search for.

Source Code Browser

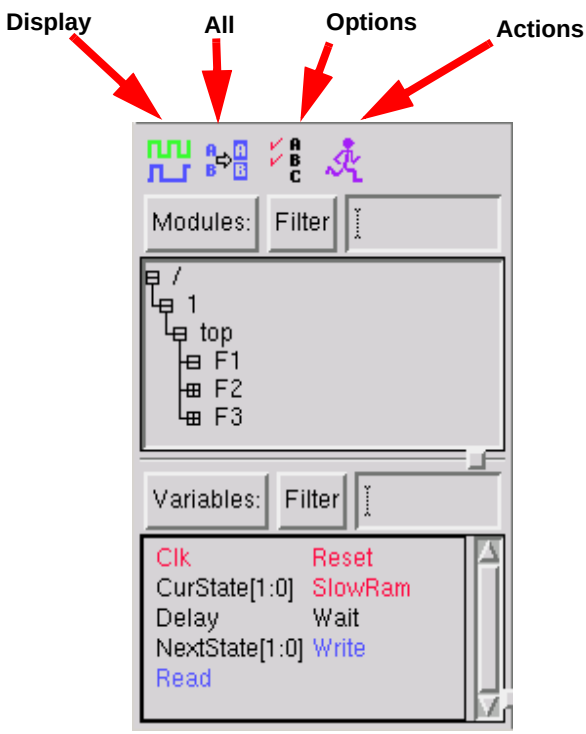
The Hierarchical Browser and Signal Selector within the Source Code Browser are located just to the left of the Source Code Window.

You can use these windows to:

- Open the design hierarchy.
- Select a level of hierarchy.
- Select signals to display in the Waveform Window (See Figure 116).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 116. Source Code Browser



The Source Code Browser tool bar includes (See Figure 115):

- **Display** — Displays the selected (highlighted) signals in the Waveform Window when toggled on.
- **All** — Select (highlight) all signals in the signal browser when toggled on.
- **Options** — Displays a menu containing the following options: **All Files**, **Overlay Files, Hierarchy, Show Filenames, Input, Output,** and **Split Seeps and Memories**.
- **Actions** — Displays a menu containing the following options: **Copy**, **Text**, **Explode Bus**, and **Find Signals**.
- **Help** — Brings up the help dialog box.
- **Module** — When toggled on, a left click on the hierarchical window will load that module into the Source Code Window.

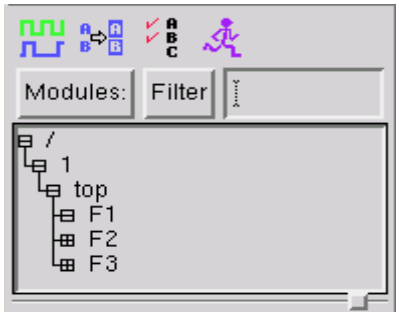
Hierarchical Browser

In the Hierarchical Browser the hierarchy is shown with a /1 at the very top, and below this the module name for the top hierarchical level of this design is displayed. To the immediate left of each module name is a "+" sign, indicating that this hierarchical level has not yet been expanded.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

When expanded or opened at this level the "+" sign will be turned into a "-" sign and the hierarchy level one level below this level will be opened (See Figure 117).

FIGURE 117. Hierarchical Browser



You can also click the Filter button, and enter the name of a module or group of modules to display. For example, you can enter **F*** to display the F1, F2, and F3 modules.

1. Click on the "+" sign to the left of the number 1 in the Hierarchical Browser. The next level below now displays the module name for the top hierarchical level of this design, in this case the module "top".

   - To expand or open the hierarchy at any level, click on the "+" sign, it will be turned in to "-" sign and display one hierarchy level below your current level.

   - To close the hierarchy at this level click the "-" sign.

2. Click on the name at any hierarchical level as this will cause the signal names to be displayed in the Signal Selector. The module name selected will be highlighted (reverse video) indicating that it is the signal names of this level that will be displayed in the Signal Selector.

Signal Selector

To display any signal from the source code in the Signal Selector proceed as follows:

1. In the Source Code Browser, select the module containing the signal to display.

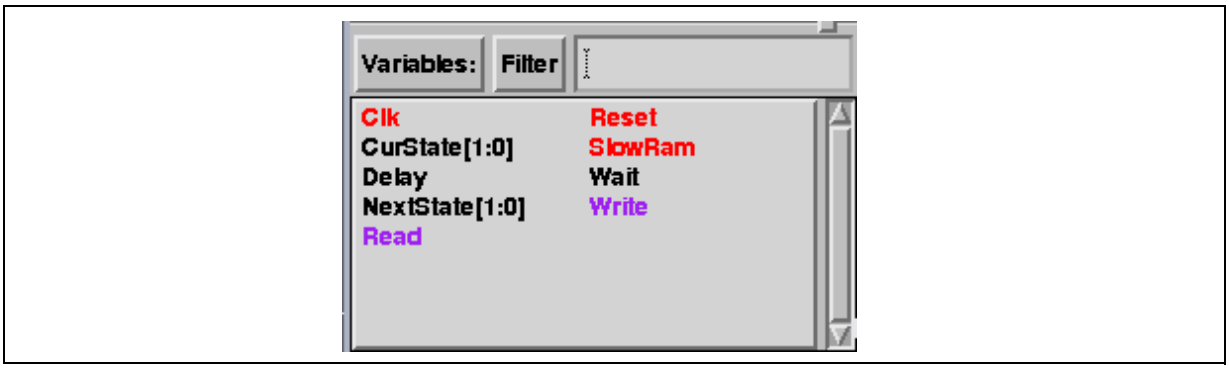
2. Click the **Variables** button to display all signals associated with the selected module.

3. Click the signal to display in an Waveform Window.

You can also click the **Filter** button, and enter the name of a signal or group of signals to display. For example, you can enter **Clk*** to list all clock (Clk) signals.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 118. Signal Selector displaying signals



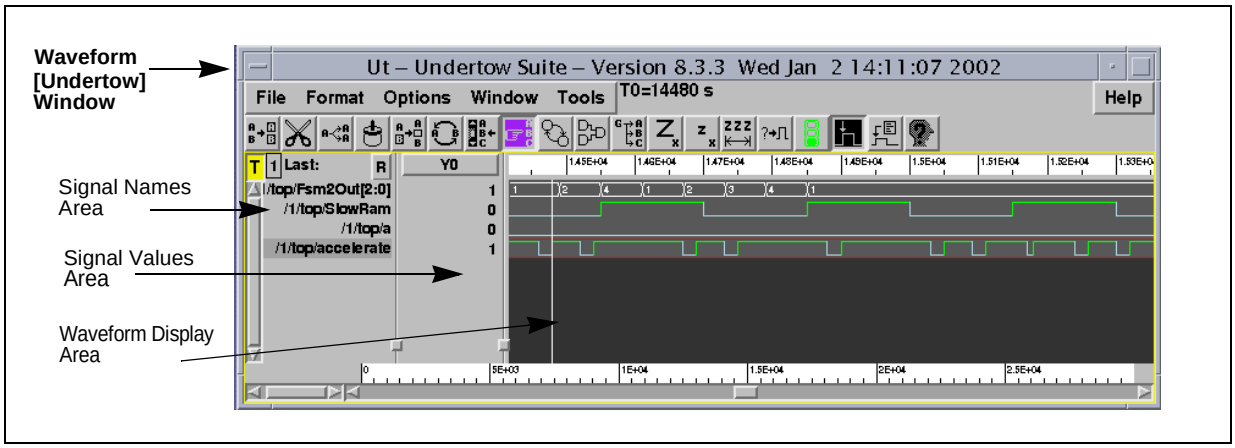
In the Signal Selector, the signal names will be color coded as follows:

- **Red** — If the signals are inputs.
- **Blue** — If the signals are outputs.
- **Black** — If the signals are neither inputs nor outputs.
- Purple — If inputs and outputs are combined (See Figure 118).

Waveform Window

The Waveform Window, called Undertow, displays the waveform for the design. Waveform file can be in VCD format, "Fast" files, or files that the simulator created using PLI or FLI.

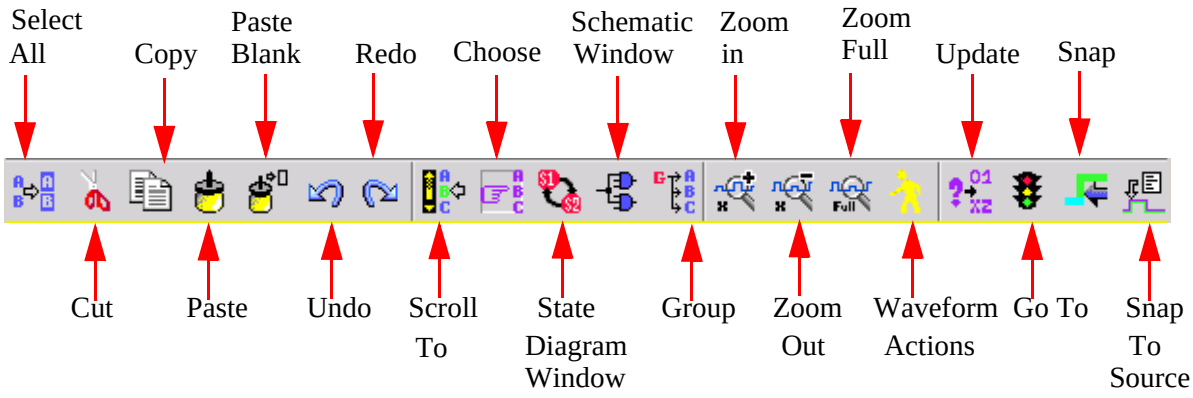
FIGURE 119. Waveform Window



Multiple waveforms can be displayed in one or more panes within the Waveform Window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 120. Waveform Window tool bar



The Waveform Window tool bar includes:

- **Select All** — Select all of the signals that are currently displayed.
- **Cut/Copy/Paste/Paste Blank** — Editing functions for the signals displayed in the Waveform Window.
- **Undo** — Undoes the last action performed in the Waveform Window.
- **Scroll To** — Opens a dialog box that you can use to build a list of signals.
- **Choose** — Toggles on or off viewing the Source Code Browser.
- **State Diagram Window** — Brings up the State Diagram Window.
- **Schematic Window** — Brings up the Schematic Window.
- **Group** — Appends or removes the group window at the right side of the Waveform Display Window.
- **Zoom In/Zoom Out/Zoom Full** — Changes the viewing scale for the Waveform Display Area.
- **Update** — Performs a screen refresh. Use this icon to display a waveform in real-time mode, so you can load in your latest data.
- **GoTo** — Pops up as dialog box for navigating to a specified point of interest.
- **Snap/Snap Source** — Snaps the selected cursor to the nearest edge of a selected signal, snap source synchronizes with the source code in the Source Code Window.

 Copyright Veritools 2004 All Rights Reserved

- **Help** — Brings up the help dialog box.

Signal Names Area

To view the whole signal name within the Signal Names Area, hold down the right button on any signal. Signals may be moved around within the list.

Signal Values Area

This column shows a value for each signal at the Time Cursor 0 (T0) position.

Waveform Display Area

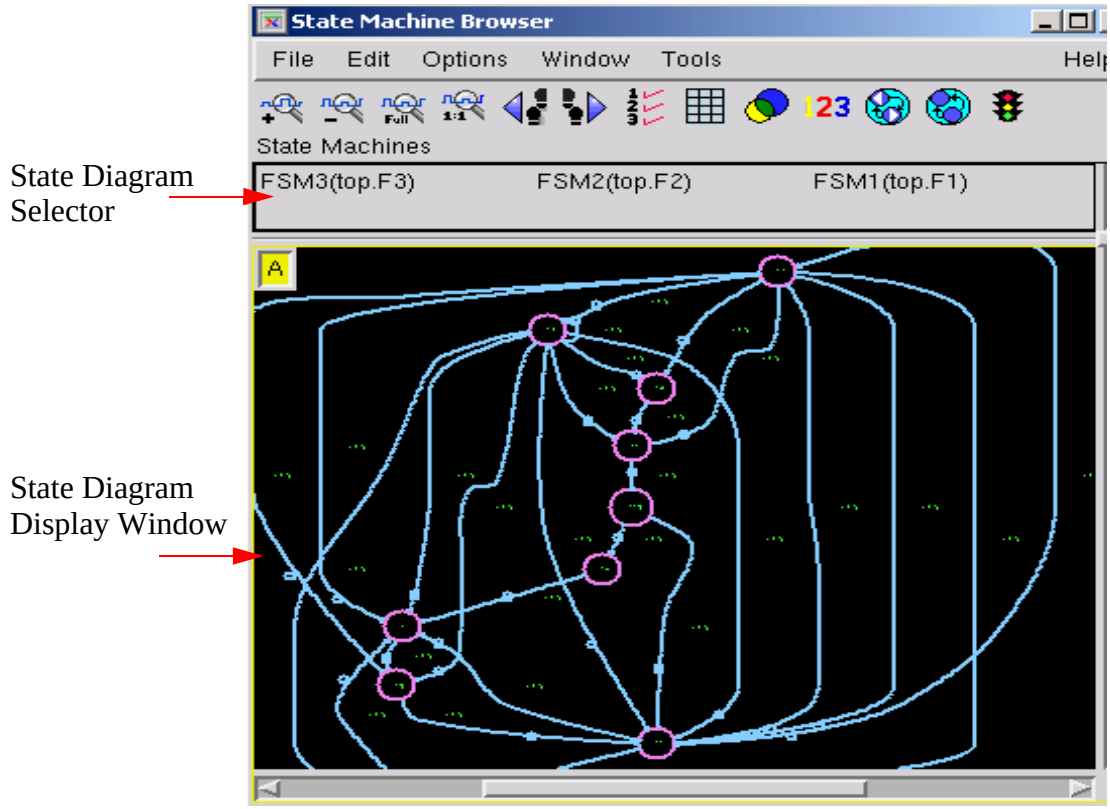
The Waveform Display Area is the main portion of the Waveform Window. To modify how many waveforms initially display on any screen, or the pixel height of any signal, edit the Undertow file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

State Diagram Window

The State Diagram Window displays state diagrams for the current design (See Figure 121). The State Diagram Window also includes a number of state diagram analysis tools.

FIGURE 121. Selecting state diagram for viewing

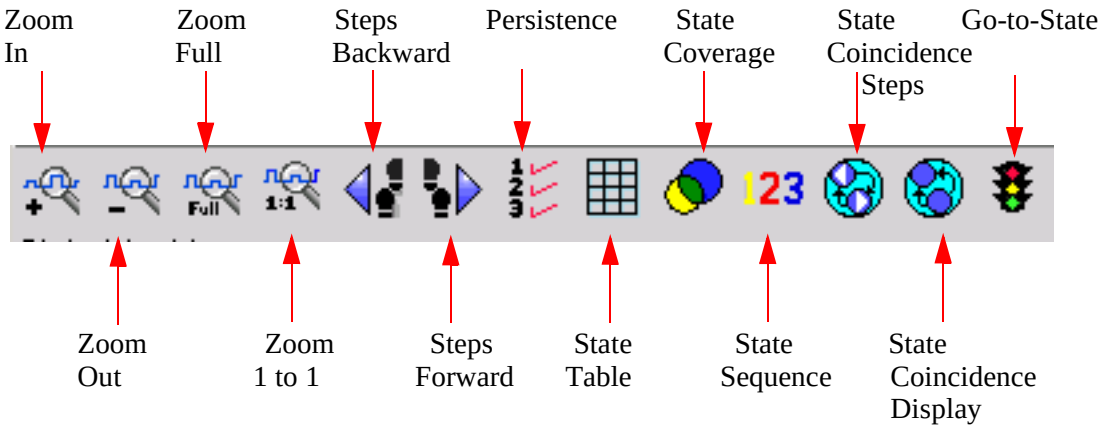


The State Diagram Window has two sections, the State Diagram Selector, at the top of State Diagram Window, and State Diagram Display Window. Selecting any state diagram in the selector will show that state diagram in State Diagram Display Window (See Figure 121).

You can create as many panes as you want in a State Diagram Window, and load any state diagram into any pane. When a state diagram is in *focus* in the State Diagram Window, you can use the window's icon controls to step forward or backward through the state diagram (See Figure 122).

 Copyright Veritools 2004 All Rights Reserved

FIGURE 122. State Machine Window tool bar



The State Diagram Window tool bar offers the following functions (See Figure 122):

- **Zoom In/Zoom Out/Zoom Full/Zoom 1 to 1** — Changes the viewing scale for the Waveform Display Area.
- **Steps Backward/Steps Forward** — Use these controls to step forward or backward by one time unit. This operation requires a trace file.
- **Persistence** — Shows which states have been entered.
- **State Table** — Shows what states are possible transitions from other states.
- **State Coverage** — Shows both what states have been covered, or traversed, and what paths have been traversed.
- **State Sequence** — Allows the user to see exactly how the state diagram sequence occurs.
- **State Coincidence, Display** — Shows whether any selected state in the state diagram is in focus, and what state sequences the other state machines are going through.
- **State Coincidence, Step** — Steps through the state diagram.
- **Go-to-State** — Select a state in the state diagram, then click this icon to go to the state in the Waveform Window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

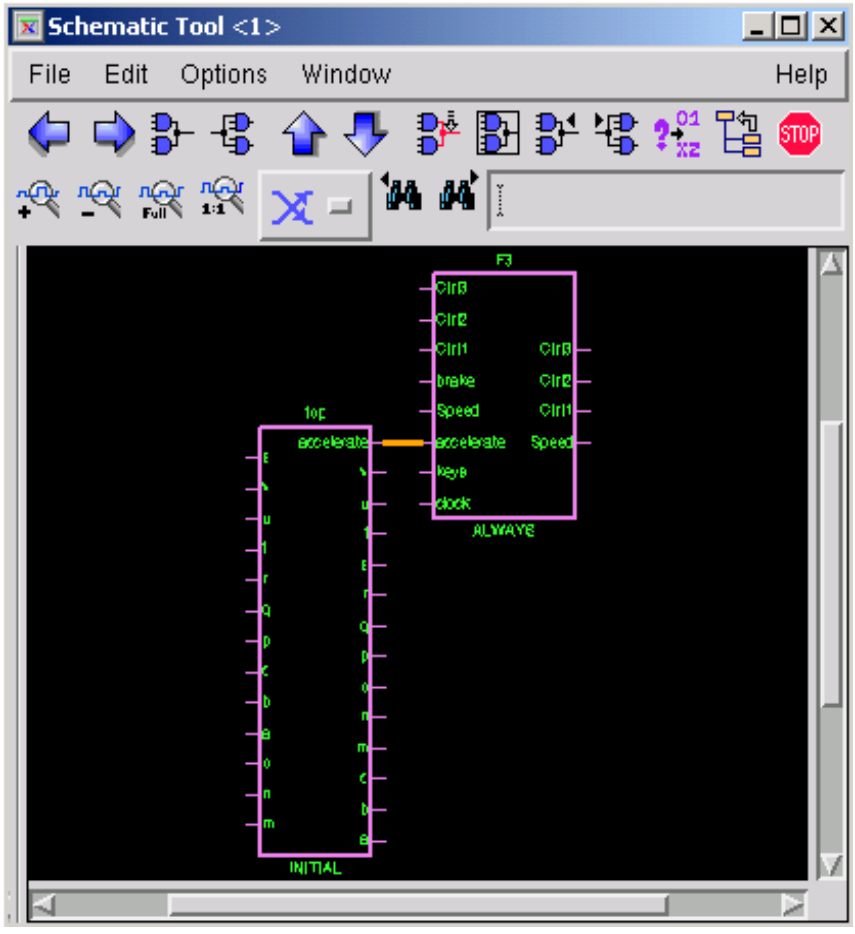
Schematic Window

The Schematic Window displays the Register Transfer Level (RTL), Gate, or RTL/Gate schematic of your design. You define the starting level of the schematic to display.

You can drag and drop any signal name into the Waveform Window to view the waveform for this signal, or into the Source Code Window to find this signal in the source code.

All of these windows are synchronized in time, when you use the Undertow Suite. All of these widows support drag-and-drop, so you can effortlessly move data between the different windows. This feature lets you correlate the view in one window with the view in any other window (See Figure 123).

FIGURE 123. Schematic Window displaying variable *accelerate*



To display only a logic path traced back from an output pin, double click on the output pin.

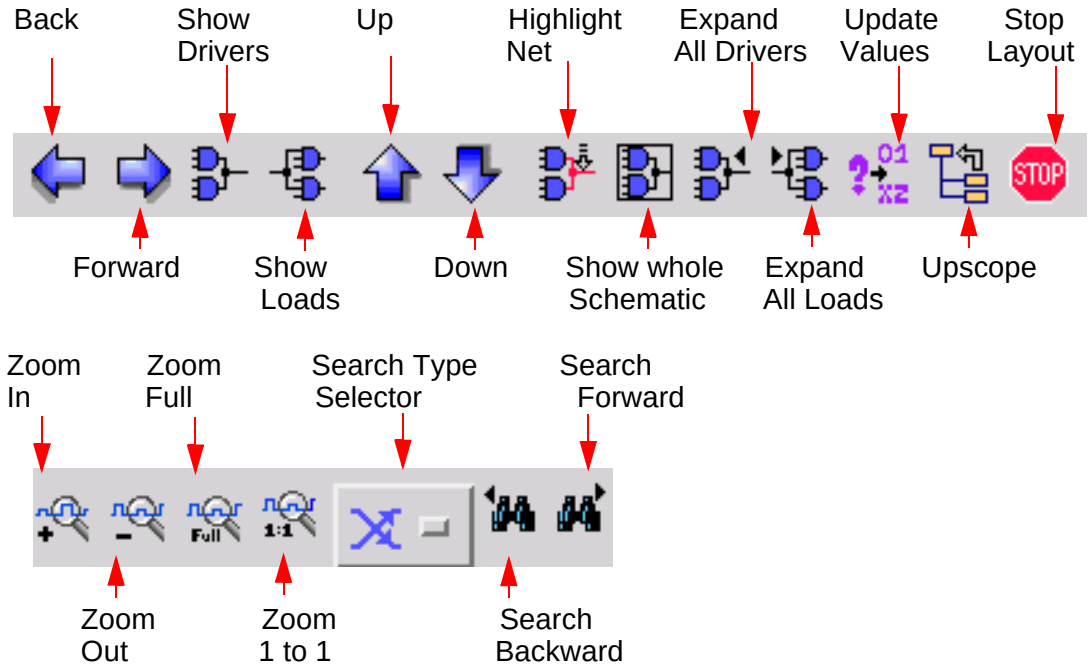
For every signal shown on the schematic, the **Schematic** window annotates the signal values at cursor time "0". If you display the entire schematic, you can graphically show where this signal is connected into the rest of your design:

1. Click the **Highlight Net** icon on the gate browser window.
2. Drag and drop a signal name from the **Waveform** window, **Source Code** window, or **Signal Selector**.

   The **Schematic** window highlights this signal in orange, so it is clearly visible, even in a very dense schematic drawing. This window also shows all drivers and loads connected to this signal, as attached to this highlighted net.

Figure 124 show the icons on the Schematic Window tool bar.

FIGURE 124. Schematic Window tool bar description



The State Diagram Window tool bar offers the following functions:

- **Zoom In/Zoom Out/Zoom Full/Zoom 1 to 1** — Changes the viewing scale for the Schematic Window.
- **Show Drivers/Show Loads** — Shows the drivers and loads in the Schematic Window.

- **Up/Down** — The Up and Down icons are enabled when a signal is selected. Selecting one of these icons moves the zoomed in schematic view to all signal connection in the schematic for viewing.
- **Highlight Net** — Displays selected net in bright orange.
- **Show Whole Schematic** — Shows the entire gate/RTL schematic in the window.
- **Expand All Drivers/Expand All Loads** — Expands all drivers one level above or all loads one level below.
- **Stop Layout** — Stops the generation of large schematics.
- **Search Selector** — Choose whether to search for a Transition, a Leading (rising) Edge, a Trailing (falling) Edge, a Value, or a String.
- **Search Backward/Search Forward** — Search forward or backward for the item chosen in the Search Selector.
- **Search String** — Use this text box in conjunction with the Search Selector items value and string. Type in the value or text string to search for.

Starting Undertow Suite

You can start Undertow Suite in three different ways:

- **use a command line** .
- **load files and simulator type using the GUI**
- **use a project file**.

Note:Starting Undertow suite using a project file can be done only if a project file is saved anytime during a previous session.

1. In the Unix .cshrc file, make sure that the PATH variable includes the **UT_ROOT_DIR** directory (the directory that holds the **ut** executable).

   When you run interactive simulation, you should normally start the Undertow Suite in the same directory where you run your Verilog or VHDL simulator.

2. Set the following environment variables, as described in Section , "Environment Variables in a Unix Environment":

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **UT_ROOT_DIR**
- **LM_LICENSE_FILE or VERITOOLS_LICENSE_FILE**
- **UT_WORK_DIR**

3. To start the Undertow Suite, type:

   %ut -iv

   The Undertow **Waveform window** opens first, then the **Source Code Window**, and finally the **Source Code Browser** opens on the left side of the Source Code Window.

Loading files from command line

You can load files from the Undertow Suite menu options (see Section , "Loading Files From GUI Menu"), or via the command line. Table 47, "Undertow Suite Command Line Options" lists the most common command line options used with Undertow Suite.

TABLE 47. Undertow Suite Command Line Options

Option	Description
-f <*filename*>	Verilog or VHDL source file to load. You can load multiple source code files. Use **-f** in front of each source file name.
-sigfile <*signal filename*>	Signal file that your simulator generates.
-tracefile <*trace filename*>	Events data file.
-prj <*projectFileName*>	Undertow Suite Project File.
-map <map_file_name>	To map different hierarchical names in the source code parsing and dump files. The map file is an ASCII file with one hierarchical mapping per line. A line has the format: source.hierarchical.name=dump.file.hierarchical.name

The following are examples of using command line options:

ut -iv -f fsm1.v -f fsm2.v -f fsm3.v -f top.v

ut -iv -sigfile verilog.dump -f vfiles

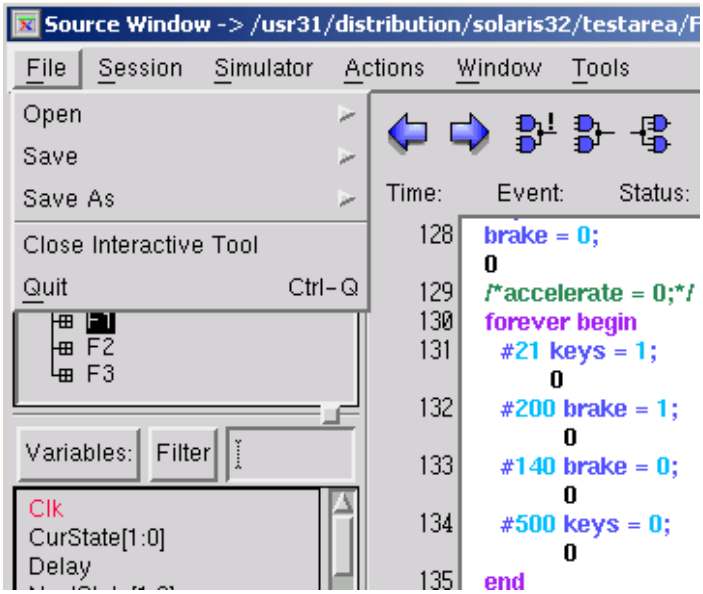
ut -iv -prj fsm.prj

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Loading Files From GUI Menu

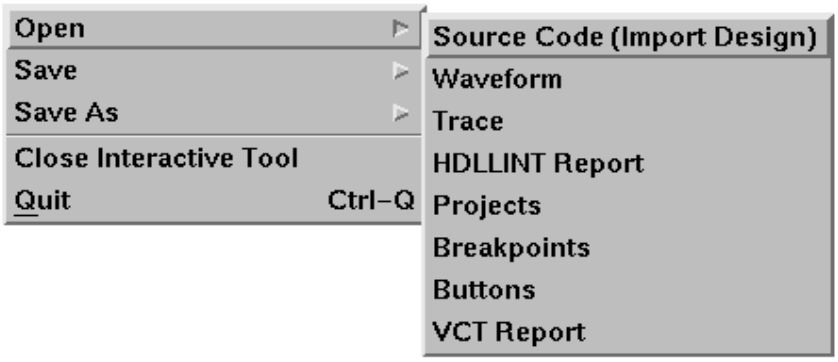
When you type **ut -iv** from the command line, Undertow Suite starts without loading any files.

FIGURE 125. Opening Files Menu



You can use the **File => Open** menu to open and load input files in Undertow Suite.

FIGURE 126. File Open Options



Modes of Operation

Undertow suite can be run in either of two modes: Batch mode, Interactive mode Batch mode implies no active usage of simulator license It uses the files created during an earlier simulation session, to . Interactive mode, on the other hand, runs a live-simulation

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Batch Operation of Undertow Suite

To use Undertow suite in Batch mode using the GUI, perform the following steps:

1. Using the **GoTo** menu in the Source Code Window, select your source code files.

   File **=>** ***Open*** **=>** ***Source Code*** (Import Design)

   The file selector opens.

2. Open the directory that holds your source code files.
3. Select one source code file at a time, and press **Add File To List** for each file.
4. Repeat Step 3 to select all of the source code files used in your design.
5. The lower text window lists all of your files.
6. To load your design, click **Apply**.

You can edit this window, to add any commands that the simulator might require before it can interpret these files.For instance, if there are many source-code files and the names of them all are in a file named source.vc, you need to:

1. Select the source.vc file and add it to your list.
2. Add **-f** (followed by a single space) in front of the file name, source.vc

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Be sure that you include a space between the **-f** and the file name.

This example tells both Undertow Suite and the simulator that all of your files are listed in the source.vc file.

3. After you select the files that you want, click **Apply**.

   Undertow loads the files into a database, and parses the files, for use by the Undertow Suite.

4. To select your waveform file, in the Undertow waveform window, go to:

   File => Open/Close File

   The file selector opens.

5. Select your waveform file. You can use any of the following file types to batch operate the Undertow Suite:

   - VCD files - Standard Verilog XL dump file
   - .fast files - Optimized File created by the **$vtDump** PLI routine
   - .sig files - Highly compressed files created by the **$vtDumpvars** PLI
   - .wav files - ModelTech VHDL, pre 5.3
   - .wlf files - ModelTech VHDL dump file

Interactive Mode

Undertow suite can be used in Interactive mode on two types of Simulators: Non-Compiled and Compiled Simulators.

- Interpreted simulators are those wherein the source code files and the simulator commands are sent to the simulator during simulation.
- Complied simulators are those wherein the source code files are used in the compile step before simulation, and the interactive simulator commands are sent to the simulator during simulation

Example of Non-Compiled simlator is Verilog-XL. Some of the Compiled simulators include Cadence Verilog NC, Synopsys VCS, Fintronic FinSim, Frontline SimLine, Modeltech Verilog, Modeltech VHDL Optimizing Tool (FLI), Modeltech VHDL VCD File. To use Undertow suite in Interactive Mode using either type of Simulator, in the GUI mode, perform the following steps

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

IUsing Non-Compiled Simulators

Either before or after you select your source code and waveform files, you can select the simulator type, and define the name of the simulator that you want to use.

1. From the Undertow menus, select **Sessions => Source Code Files/Simulator Options**.

   The Source Code Files/Simulator Options window opens.

2. Using the **Simulator Type** pull down, press, pull down, and select the simulator type to use:

   Cadence Verilog XL

3. In the **Simulator Executable** window, enter the name of the simulator.

   The text that you enter in this window is relative to the directory where you started the Undertow Suite, unless you specify the full path for the simulator.

4. Open your source code files and waveform file, as described in Section , "Batch Operation of Undertow Suite".

5. To run the simulator, select **Session=>Simulator=>**. Choose the appropriate option from amongst the avalable options as shown below

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Using Compiled Simulators

To run interactive simulation, using a complied simulator, you must both select a simulator type and define the name of the simulator that you want to use, before you select any source code files.

To select the simulator type:

1. From the Undertow Suite menus. select **Session => Source Code Files/Simulator Options**
2. Using the **Simulator Type** pull down, pull down and select the compiled simulator type to use:

   Cadence Verilog NC

   Synopsys VCS

   Fintronic FinSim

   Frontline SimLine

   Modeltech Verilog

   Modeltech VHDL Optimizing Tool (FLI)

   Modeltech VHDL VCD File

3. In the **Simulator Executable** window, enter the name of the simulator.

   The text that you enter in this window is relative to the directory where you started the Undertow Suite, unless you specify the full path for the simulator.

4. Specify the simulator commands to send to the simulator when you run simulation. Type these commands into the Simulator Command Line window.
5. Open your source code files and waveform files as described in the section."Batch Operation of Undertow Suite"
6. To run the simulator, select Session=>Simulator=>. Choose the appropriate option form angst the following

Using a Command Line to Start Undertow Suite

The Undertow Suite lets you start the software, and specify all necessary files, from a command line. The specific commands you enter on the command line depend on whether you are running

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

your simulator in batch mode, or interactive mode. Your command line options are also affected by what type of simulator you are using, either non-compiled or compiled.

Batch Simulation

Batch simulation means that you have already run your simulator and you have produced the files that you need. You can then use the Undertow Suite to debug your source code, without using a simulator license that someone else might need.

To use the command line for batch simulation, you need to specify only the source code files that you used in simulation, and any other files that you need for debugging. In batch mode, the command line uses the following format:

%ut -iv <*source_code_files*> <*other_file_types*> <*file_name*>

The fields in this command are as follows.

source_code_files

The source code files can be a list of files, or a single file that holds the names of other files. These files can be Verilog files, VHDL files, or a combination of Verilog and VHLD files.

For example, if your design used files named:

top.v
fsm1.v
fsm2.v
fsm3.v

then these are the file names listed in the *source_code_files*.

If all of these files are already listed in a file name *source_code.vc*, then you can specify the *source_code_files* as:

-f source_code.vc

Other_file_types

-sigfile *name_of_signal_file*
You can specify any of the following types of signal files:
VCD file
Fast file
Optimized file from PLI
Optimized file from FLI
.wav file form Modeltech 5.2 or earlier
.wlf file from Modeltech 5.3 or later

-tracefile *trace_file_name*
The trace file records the execution of each line of source code.

Several types of Verilog simulators (such as VCS, Verilog XL, and FinSim) can produce a trace file. The trace file acts as a Virtual Simulator while you are using the Undertow Suite.

The following are examples of some command lines for batch simulation:

%ut -iv -f source.vc -sigfile fsm.sigs -tracefile fsm.trace
%ut -iv top.v fsm1.v fsm2.v fsm3.v fsm4.v -sigfile fsm.dump
%ut -iv top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl -sigfile fsm.wlf

Interactive Simulation

Non-Compiled Simulators

If you are running a non-compiled simulator, such as Verilog-XL, the command line is almost the same as the batch mode. However, in addition to all of the batch mode commands for the simulator, in this mode you can also specify the type of simulator and the name of the simulator. The command line has the following format;

%ut -iv <*simulator_type*> <*simulator_name*> <*simulator_commands*>
<*other_file_types*> <*file_names*>

The simulator types are:

-xl Cadence Verilog-XL

The *simulator_name* is the name you call your simulator when the Veritools PLI is linked in. This name is relative to where you started the Undertow Suite, and is normally the name of the directory where you run your simulator.

*Note: If you are currently running Verilog-XL, you can just add **ut -iv -xl** in front of the Verilog and Verilog commands/source code files, to start Undertow Suite from Verilog-XL.*

For example, if your Verilog command line is as follows;

%verilog -f source_code.vc other_simulator_commands

then place **ut -iv -xl** in front of this command, as follows;

%ut -iv -xl verilog -f source_code.vc other_simulator_commands

To automatically open additional files, add the *other_file_type* and *other_file_name* options, after the simulator source code files and simulator commands. For example:

%ut -iv -xl verilog -f source_code.vc *other_simulator_commands*
-sigfile *signal_file_name* -tracefile *trace_file_name*

Other_file_types

 Copyright Veritools 2004 All Rights Reserved

-sigfile signal_file_name

You can specify any of the following types of signal files:
VCD file
Fast file
Optimized file from PLI (See Chapter 6)

-tracefile trace_file_name

The trace file records the execution of each line of source code.

Note: Make a trace file only when you want to use the "virtual simulation" feature. Creating trace files slows down your simulation.

Compiled Simulators

Many simulators are *compiled* simulators, including VCS, Verilog, NC-Verilog, Finsim, and Modeltech. The following are simulator types and their corresponding command line options:

-vcs	Synopsys VCS Simulator
-nc	Cadence Verilog-NC
-modeltech	Modeltech Verilog
-modeltechvhdl	Modeltech VHDL
-modeltechvhdlvcd	Modeltech VHDL using the VCD file
-finsim	Finsim Simulator

For waveform files, the VCS, Verilog-XL, NC-Verilog, Modeltech Verilog, and FinSim simulators can use a VCD file format, or you can use the Veritools PLI to create either a Fast File or an Optimized File.

When you run the simulator, the Undertow Suite command line automatically adjusts to the fact that these files are already compiled into a executable. The source code file names that were previously placed after the simulator name, are now placed after the **ivsimcmp** command option. **ivsimcmp** specifies the Interactive Tool simulator compile, which must be inside of double quotation marks—for example:

%ut -iv -vcs <*vcs_simulator_name*> <*simulator_commands*>
-sigfile <*signal_file_name*> - tracefile <*trace_file_name*>
ivsimcmp "-f source_code.vc"

*Note: You must include a space after the **ivsimcmp** command option, before the first set of double quotation marks.*

The following table lists the supported Simulator types, and the actions that the Undertow Suite automatically performs when you run these simulators in interactive mode:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Command Line	Simulator	Compile
-xl	Verilog-XL	No
-vcs	Synopsys VCS	Yes
-nc	NC-Verilog	Yes
-fi	Finsim	Yes
-modeltech	ModelTech Verilog	Yes
-modeltechvhdl	ModelTech VHDL	Yes
-modeltechvhdlvcd	ModelTech VHDL w/VCD output	Yes

Based on the simulator type and the command line options, the Undertow Suite issues automatic commands to the simulator:

Command Line	Command Sent to Simulator	Simulator
-sigfile xyz.sigs	$dumpfile(xyz.sigs);	Verilog-Xl/NC-Verilog, VCS, Finsim
no signal file specified	$dumpfile(ut.dmp);	Verilog-Xl/NC-Verilog, VCS, Finsim
-tracefile xyz.trace	-trace	Verilog-XL
	$vtDumpvars	Verilog-XL/NC, VCS, Finsim
	+VTCOMPRESS200	Verilog-XL/NC, VCS, Finsim
-modeltech	+VTCOMPRESS300	ModelTech Verilog
-modeltechvhdlvcd	vcd add -r *	ModelTech VHDL

For example, if you specified the following command line options:

%ut -iv -xl verilog -f source.vc -sigfile fsm.sigs -tracefile fsm.trace

then on the command line, the Undertow Suite automatically sends **+VTCOMPRESS200** to the simulator, and issues the following system tasks to the simulator:

$dumpfile(fsm.sigs);
$vtDumpvars;

The **+VTCOMPRESS200** command lets the Undertow Suite read data that the simulator writes, in real time.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

If you used this command line:

%ut -iv -modeltech vsim top -sigfile fsm.sigs -ivsimcmp "-f source.vc"

then the Undertow Suite automatically send to the Modeltech simulator, both the **+VTCOMPRESS300** command, and the following system commands:

$dumpfile(fsm.sigs);
$vtDumpvars;

For more details on linking PLI with each simulator, refer to **Chapter 9. Quick Start Guide**, sub-section titled **Linking with PLI**

Using Undertow Suite

When you type **ut -iv** (with or without command line options) to start Undertow Suite, two windows open:

- The **Waveform** window appears at the bottom of your screen.
- The **Source Code** window appears at the top of your screen.

After you load the source, trace, and signal files, as described in Section , "Using a Command Line to Start Undertow Suite" and Section , "Source Code Window", you can analyze your design, as described in the remaining sections of this chapter.

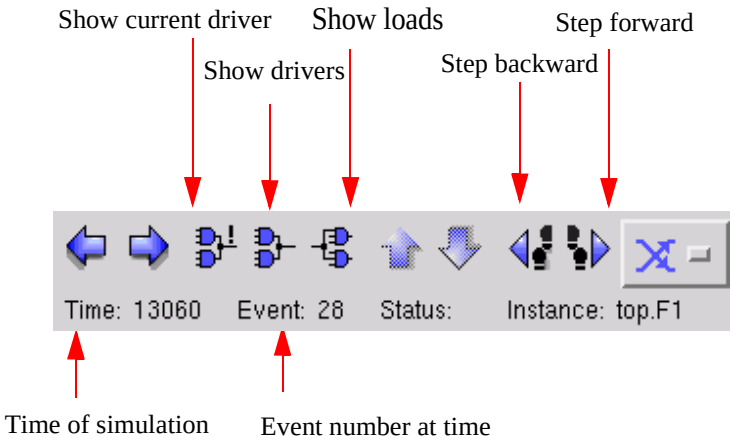
Stepping Through Source Code

After you load the waveform files that your simulator generated, you can use the Undertow Suite as a virtual simulator.

You can use the step buttons in the **Source Code Window** to step the simulator forward and backward, one event at a time. When you step the simulation, the **Source Code Window** updates to display the current line for the selected time and event.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 127. Source Code Toolbar



Displaying Loads and Drivers

To display all drivers for a selected signal:

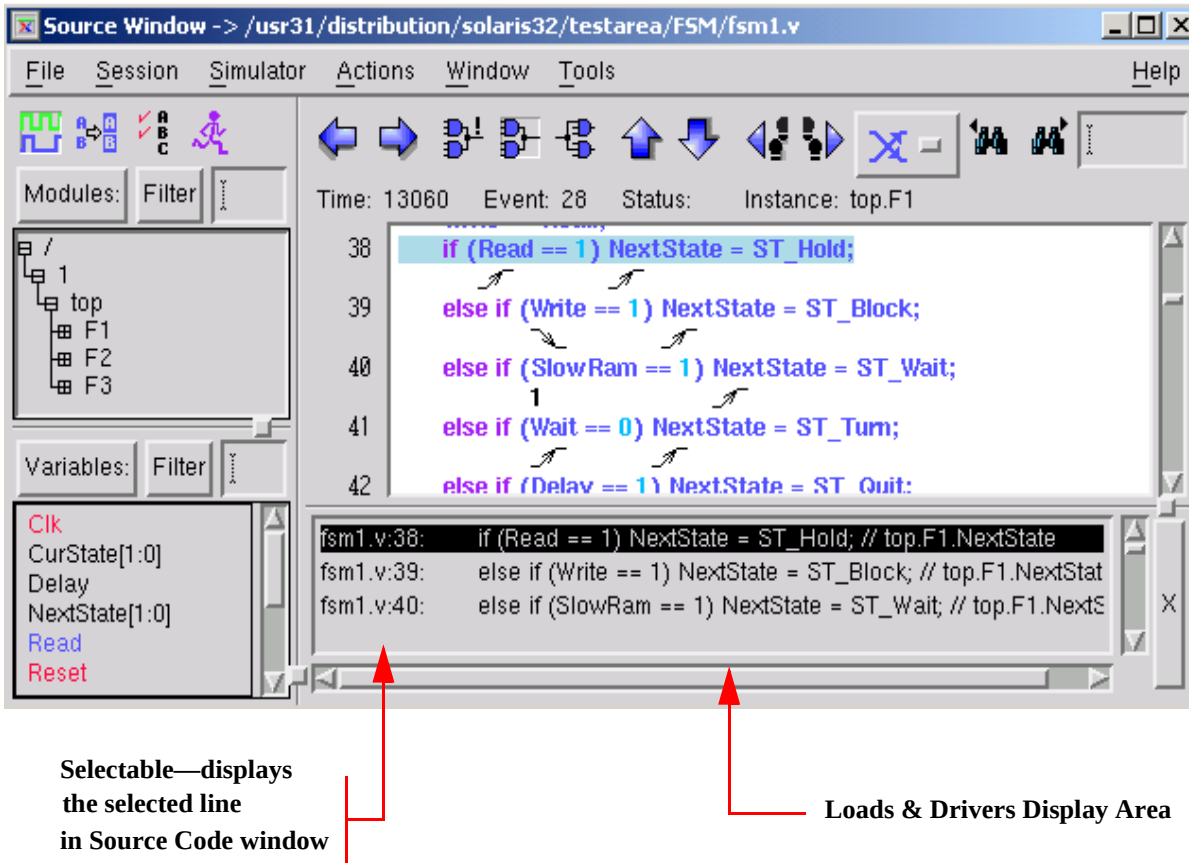
1. Toggle on the **Show Drivers** button (see Figure 128, "Loads and Drivers Display Area").
2. In the source code viewing area, click on the desired signal.

   All loads associated with the selected signal appear in a dialog box below the Source Code window.

   Note: This dialog box opens the first time you display a load or driver.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

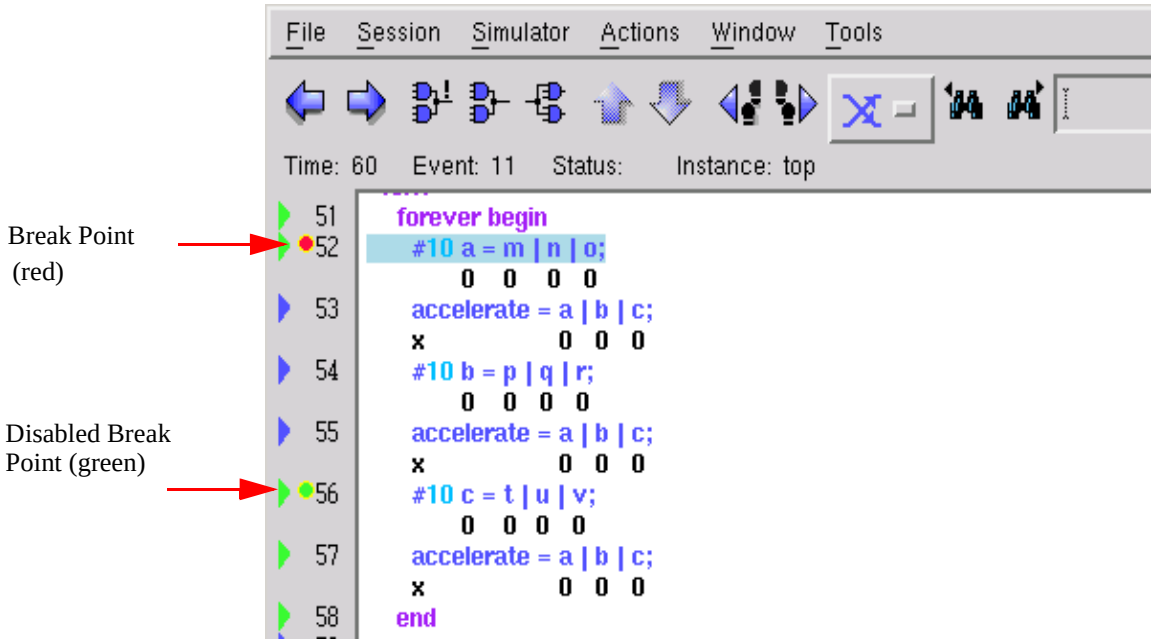
FIGURE 128. Loads and Drivers Display Area


Source Window -> /usr31/distribution/solaris32/testarea/FSM/fsm1.v
File Session Simulator Actions Window Tools Help
Modules: Filter
Time: 13060 Event: 28 Status: Instance: top.F1
38 if (Read == 1) NextState = ST_Hold;
39 else if (Write == 1) NextState = ST_Block;
40 else if (SlowRam == 1) NextState = ST_Wait;
41 else if (Wait == 0) NextState = ST_Turn;
42 else if (Delay == 1) NextState = ST_Quit;
/
1
top
F1
F2
F3
Variables: Filter
Clk
CurState[1:0]
Delay
NextState[1:0]
Read
Reset
fsm1.v:38: if (Read == 1) NextState = ST_Hold; // top.F1.NextState
fsm1.v:39: else if (Write == 1) NextState = ST_Block; // top.F1.NextStat
fsm1.v:40: else if (SlowRam == 1) NextState = ST_Wait; // top.F1.NextS
Selectable—displays the selected line in Source Code window
Loads & Drivers Display Area

Using Breakpoints in the Source Window

To set graphical breakpoints, click the left mouse button in the left-most field of the **Source** window, between the blue and green execution markers and the Verilog code line numbers.

FIGURE 129. Break Points


File Session Simulator Actions Window Tools
Time: 60 Event: 11 Status: Instance: top
Break Point (red)
Disabled Break Point (green)
51 forever begin
52 #10 a = m | n | o;
0 0 0 0
53 accelerate = a | b | c;
x 0 0 0
54 #10 b = p | q | r;
0 0 0 0
55 accelerate = a | b | c;
x 0 0 0
56 #10 c = t | u | v;
0 0 0 0
57 accelerate = a | b | c;
x 0 0 0
58 end

Left mouse button, sets the breakpoint, which displays in red.

Left mouse button on red breakpoint, disables the breakpoint; the breakpoint indicator changes from red to green.

Left mouse button plus control key, removes the breakpoint.

Left mouse button plus shift key on red breakpoint, makes the breakpoint one shot.

Execution Information in the Source Window

Click the right mouse button over any line of code, to view execution information for that line. The popup box shows the following information about the selected code:

- Line number.
- Number of times that the selected line has been executed.
- Current execution status.
- Breakpoint status.
- VCT Branch status—lists the total number of branches (summed across all **if** and **case** tests), the total coverage, and the percent of coverage.

 Copyright Veritools 2004 All Rights Reserved

- VCT Multi Branch status—contains multi-branch coverage for all **if** tests. For each **if** test, the popup box lists the line number of the **if** statement, terms, percent covered, and time covered.

Using the Waveform Viewer

You can use Undertow Suite to:

- View waveforms.
- Snap to any edge of any signal, and see the exact line of source code that caused this transition.
- Run virtual simulation.
- View both the schematic and state diagram for your design.

All views are synchronized in time and value to the **Waveform** window. You can drag and drop elements from one view to another window view, to see different views of the same element.

Displaying waveforms

Module Browser

To view waveforms:

1.Open the **Source Code Browser** window.

2.To move one level down in the hierarchy, click the left mouse button on the upper-most + sign.

3.When signal names appear in the **Signal Selector** window, you can select the module name at this level

The module name turns black, indicating that this level is selected. Any signals associated with that module appear in the **Signal Selector** window, which is the window below the **Source Code Browser** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Snapping To The Exact Line of Source Code

snapto source

To snap to the specific line of code that caused any transition on any edge in the Undertow **Waveform** window:

1.Click the **Snap/Source** icon, located in the toolbar of the **Waveform Viewer**.

2.In the display area of the **Waveform Viewer**, select an edge on a signal.

The **Source Code** window displays the exact line in your source code that caused the selected event.

- If you are running Undertow Suite with a trace file, the **Snap/Source** shows the exact line of code.
- If you are running Undertow Suite without a trace file, the **Snap/Source** shows the assignments for this transition, and highlights the first transition. To see the other assignments, click the **Up/Down Arrow** icon, which is located at the top of the **Source Code** window.
- You can also enable a results window, ***Session*** **=>** ***Enable Results window***, to see all of these assignments at once.

Using The Schematic Viewer

You can use Undertow Suite to see a graphic schematic view of your design:

- RTL design.
- Gate Design.
- RTL mixed with gates.

To open the schematic window:

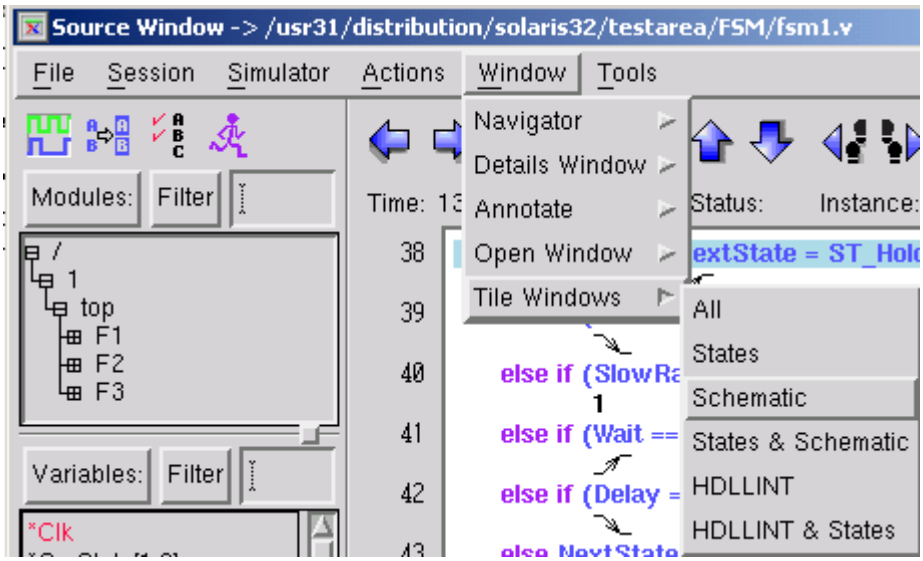
1. In the **Source Code Viewer** menus, select the **Window => Tile Windows** submenu.
2. From this submenu select either **Schematic** or **States and Schematic.**

   This second option also opens the **State** window, as well as the **Schematic** window.

Note: Another way to open the Schematic window is to click the right mouse button on a module in the ***Source Code*** *window.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 130. Starting Schematic From Source Window



Viewing Modules In Schematic Viewer

To load the **Schematic Viewer**:

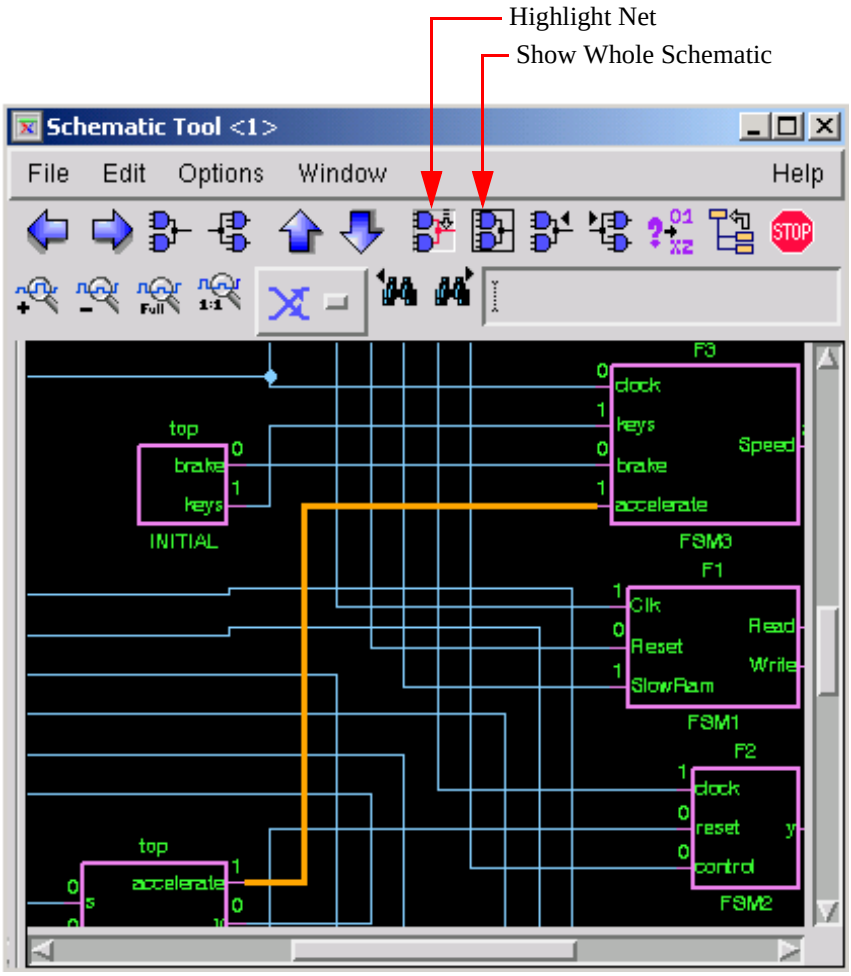
1. In the **Module Browser** of the **Source Code Viewer**, select a source module.
2. Drag and drop the selected module into the **Schematic Viewer**.

*Note: Another way to load the **Schematic Viewer** is to right-click on the module name. The **Schematic Viewer** automatically opens in* untiled *mode. This mode displays the top level of the selected module.*

- The left side of a displayed module shows the module's inputs, and the right side shows the outputs, as illustrated in Figure 131, "Schematic Window".
- To see a detailed view of the module, double left click on the module displayed in the Schematic window (in Figure 131, click inside the **top module** box). The schematic window displays the next scope, as shown in Figure 131, "Schematic Window".

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 131. Schematic Window



Finding Connections

To display a signal's connections in the **Schematic** window:

1. In the toolbar of the **Schematic** window, toggle on the **Highlight Net** button.
2. Select a signal from the **Source Code** window, the **Signal Browser**, or the **Waveform** tool.
3. Drag and drop the selected signal into the **Schematic** window.

   An orange line displays the connections associated with the signal.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Showing Multiple Scopes

To view multiple levels of hierarchy:

1. Click the left mouse button on the upper-most + sign in the **Source Code Browser** window.

   The hierarchy opens down one level.

2. Right click on a module you want to view.

   The top-level view of this module opens.

3. Double left click to show the next lower level of modules inside the selected block.

HDL Analyzer

Undertow draws a complete schematic of your Verilog HDL source code. Using the HDL Analyzer, a schematic is drawn with RTL primitives.

1. Here is a short description of the users interface

   a. Show schematic details

   b. Drag and drop a wire, signal or module onto the schematic window

   c. Double click on the empty area of any module, always block, continuous assignment or initial block to see the synthesized RTL in that block.

   d. If any pin inside of block is high lighted, then the schematic only traces the logic from that pin through block to opposite pins in block.

   e. If user double clicks on any pin inside of always block, continuous assignment or initial block, then just the drivers or loads on that pin are shown, logic back tracking stops at one level. Users can then use "Show all Drivers", or "Show all Loads" in order to see logic drawn on each subsequent level.

   f. Users can use Options => Create New Window to create as many schematic windows as they want and can drag and drop any signal, block or gate from one window to the next.

   g. Users can press the Back or Forward icons to see the last schematic screen back or forward.

   h. Users can press "Upscope" to see the scope one level above the scope where they are located. If they have high lighted any signal, this signal stays high lighted in the "Upscope" view also.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

i. User can drag any element from the schematic to the source code window in order to see the source code for that element, elements include wires or signals, blocks, or any gate including any RTL primitive gate.

j. Users can turn High light new on and drag any signal to schematic to see that signal high light if it is this schematic view. Users can also use the left mouse button when High Light net is on to high light any net.

k. When any net is high lighted, pressing Up and Down arrows will bring into the center of the screen one at a time the connections on this net.

Trace Back

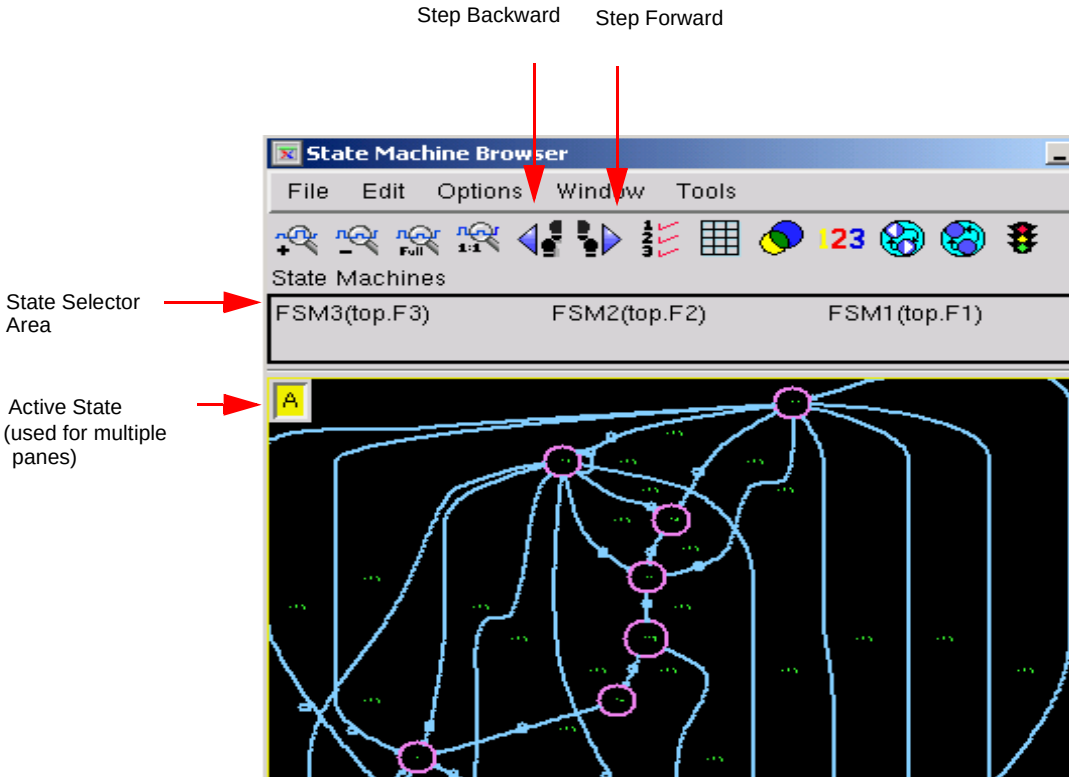
1. Users can trace any net backward or forward using the Show All Drivers or Show All Loads icons, or double clicking on the net, note the net glows when you are in a position to double click on it.

NOTE: When users have a box block to view, i.e., always, continuous assignment or initial block, clicking on the name inside dives down one level and show connection for one level of elements connected to this pin, double clicking on outside pin on box, connects net or disconnects net to this pin.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Using The State Diagram Viewer

FIGURE 132. State Window



To open the **State Diagram** window:

1. From the Source Code window, select the **Window => Tile Windows** submenu (see Figure 130, "Starting Schematic From Source Window").

2. From the **Tile Windows** submenu, select either the **States** or the **States and Schematic** menu item.

   This second option opens the **Schematic** window as well as the **State Diagram** window.

The **State Diagram** window includes a state selector area (see Figure 132, "State Window"), located at the top of the window. You can use this area to view any of the state diagrams that you extracted from your source code.

When you select a state, the *Active* pane shows that state. The Active pane is the pane where the toggle button (located in the upper left corner of each pane) is yellow. A yellow state indicates that this pane has the focus.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

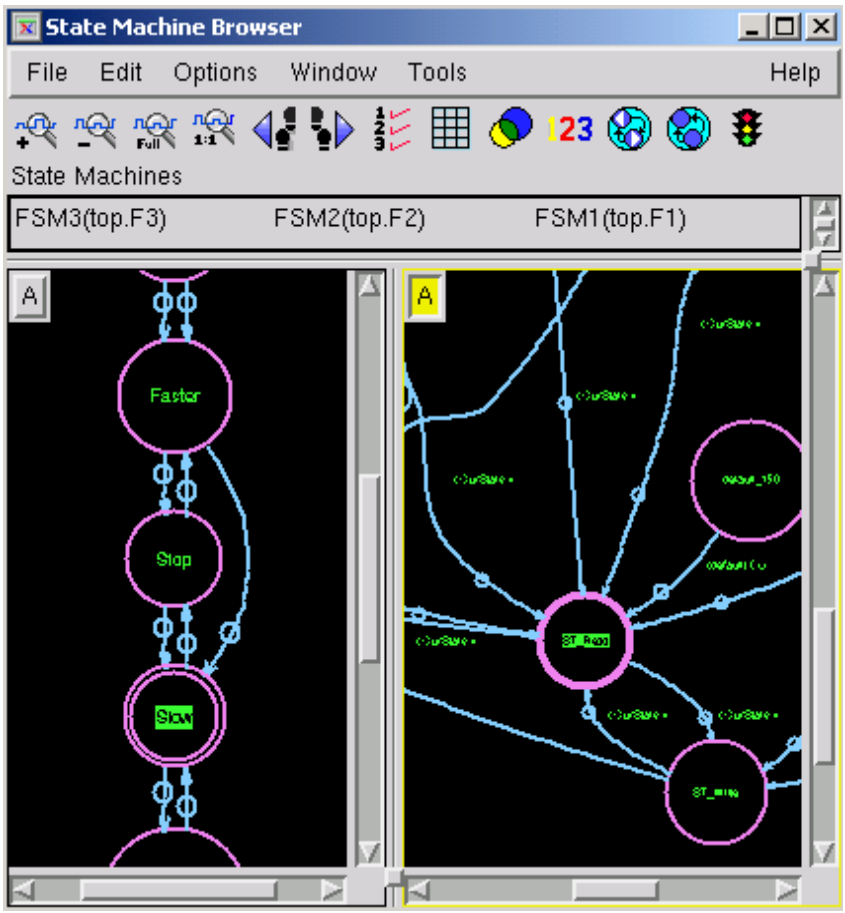
Adding Panes to the State Window

You can add as many panes as you need in the **State** window. To add a pane:

1. From the **Window** menu, select either **Add Pane Vertical** or **Add Pane Horizontal**.

   Figure 133, "State Viewer With Multiple Panes" shows a multiple-pane window.

FIGURE 133. State Viewer With Multiple Panes



State Stepping (Forward / Backward)

You can step one or more state diagrams, either forward or backward, to not only see how each state diagram works, but also to see the interaction between the different state machines.

1. Open more then one state pane.
2. To select the pane to step, toggle the **Focus** button (see Figure 133, "State Viewer With Multiple Panes").

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The **Focus** button for the active pane toggles to yellow when it has the focus.

"How To" Procedures

How To Select Source Code Files

You can use either the command line, or the source code file selector, to select the desired source code files.

To use the source code file selector:

1. From the **Source Code** window menus, select:

   File **=>** ***Open*** **=>** ***Source Code*** (import design)

   The **Select Files** window opens. This window lists directories, and files in the selected directory.

2. To select files, do either of the following:

   - Double click the left mouse button on the name of the file(s) to select, or
   - Single click the left mouse button and select **Add File To List**.

   Note: You can select more than one file.

   - If the selected file contains a list of files, insert **-f** in front of the file name.
   - If the file is a library, insert **-v** in front of the file name.

3. Click the **Apply** button.

   The Undertow Suite compiles the source code that you selected, and loads the top-level design module into your **Source Code** window.

How To Show/Locate Source Code For A Module

The **Source Code Browser** window is the little pane to left of source code window.

1. In the **Source Code Browser** window, left click the + sign, which is to the immediate left of the slash.

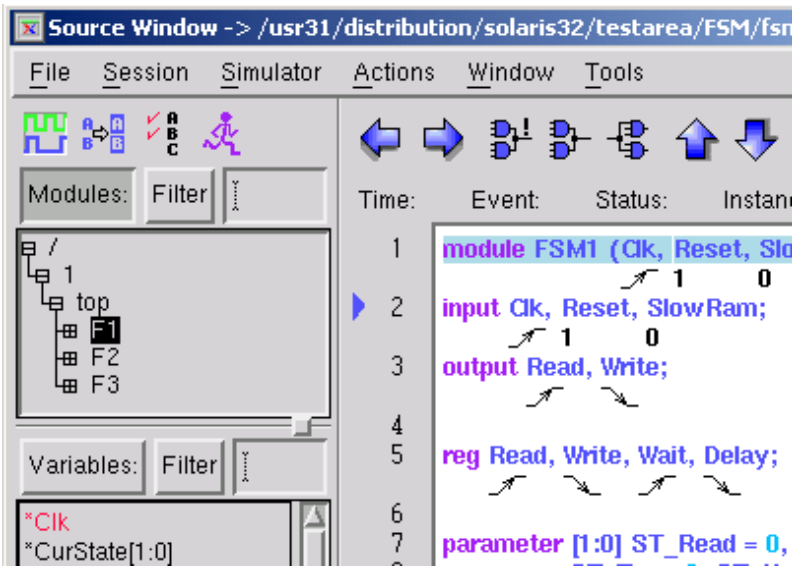
   The hierarchy expands to the level immediately below the current level.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. To show the exact source code for each module, do either of the following:

   - Use the middle mouse button to **drag and drop** the name of the module into the **Source Code** window, or
   - Toggle the **Modules** button, and use the left mouse button to **select** the module name.

   The **Source Code Browser** highlights the selected module in reverse video. The **Source Code** window displays the source code associated with this module.



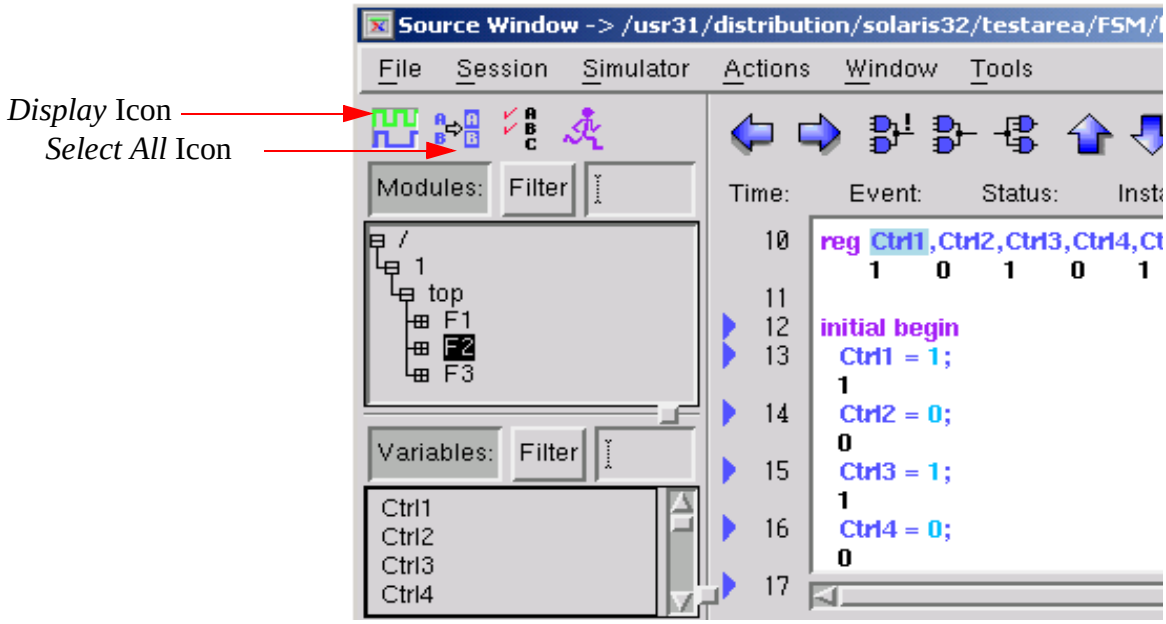
How To Find Signal Assignments/Use In Source Code

When you select a module, the **Signal Selector** pane (located below the **Module** pane) lists the signals associated with the selected module.

1. Select a signal.
2. Toggle the **Variable** button.

   The **Source Code** window shows the first occurrence of that variable (signal).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



How To Display a Signal From the Source Code Window

After you select a module in the **Source Code Browser**, the **Signal Selector** pane (located below the **Module** pane) displays the variables (such as signals) associated with that module.

You can use any of the following three procedures to display a signal in the Undertow **Waveform** viewer:

- Select one or more signals from the **Signal Selector** pane, press and hold the middle mouse button, and drag the signal(s) to the left pane of the **Waveform** window; or
- Click the **Display** icon (the icon furthest to the left in the **Source Code** window), then select one or more signals in the **Signal Selector** pane; or
- In the **Source Code** window, press and hold the middle mouse button on a signal name, drag the signal name to the **Waveform** window, and release the mouse button.

Using any of these three methods, the **Waveform** window displays the selected signals.

How To Find Source Code From the Waveform Window

1. In the **Waveform** window, click the **Snap to Source** icon.

   This icon darkens.

2. Use the middle mouse button to snap the cursor to the edge of the signal.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The Undertow Suite **Source Code** window displays the line of code that generates the selected signal edge.

Snap to Source Code

How To Start the Schematic Window

1. From the pulldown menus in the Source Code window, select one of the following three options:

   Tools **=>** ***Schematic***, or
   Windows **=>** ***Tile Windows*** **=>** ***Schematic***, or
   Windows **=>** ***States and Schematic***

   The **Schematic** window opens.

2. Do either of the following:

   - In the **Source Code Module Browser**, select a module, then drag and drop the selected module to the **Schematic** window; or
   - Click the right mouse button on the module name.

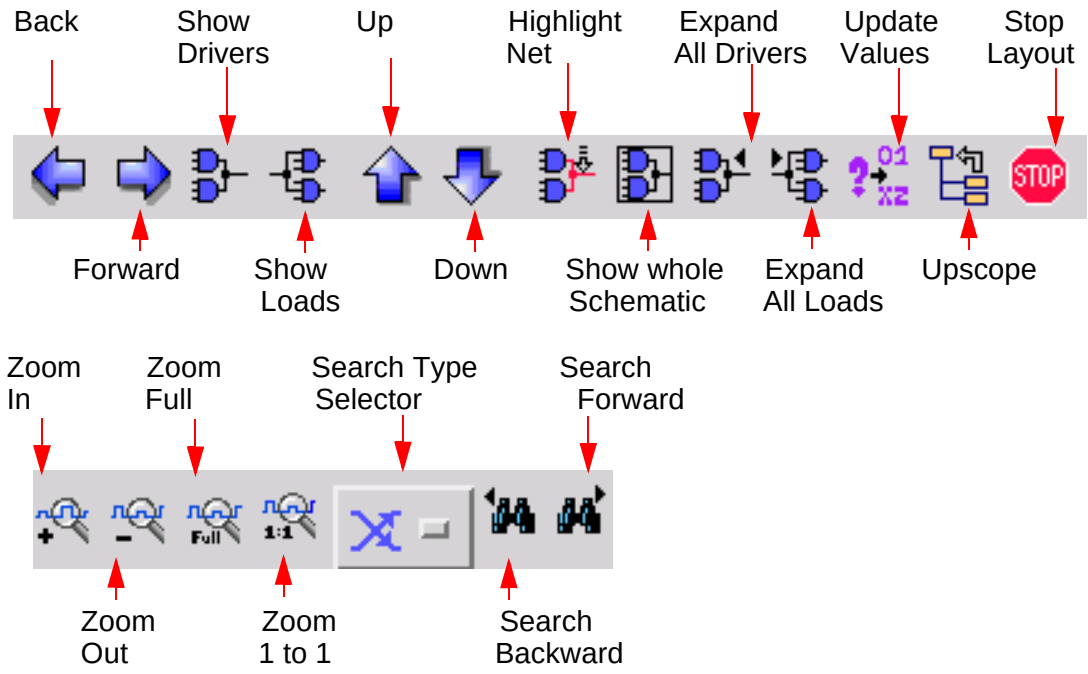
   The **Schematic** window shows the schematic view of the selected module. The input signals appear on the left side of the module, and the output signals appear on the right side of the module.

3. *(Optional)*. To display all levels of hierarchy in the schematic, select the **Show Entire Schematic** icon (the fourth icon from the right).

4. *(Optional)*. If the schematic is very large and is requiring too much time to draw, click the **Stop** icon, open a lower level of your design's hierarchy, and repeat Steps 1 and 2.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 134. Icons in Schematic Window



How To Trace A Signal In Schematic Window

1. To open the **Schematic** window, do one of the following:
   - In the Undertow **Waveform** viewer, select the **Schematic** icon, or
   - In the Undertow Suite **Source Code** window, select ***Window*** => ***Tile Window*** => ***Schematic*** from the menus, or
   - Right click on a module name to open the selected schematic in a **Schematic** window.
2. In the **Source Code**, **Choose**, or **Undertow** window, use the middle mouse button to drag and drop a signal into the **Schematic** window.

   The Schematic window displays all signal drivers.
3. To trace a signal driver in the schematic, double-click the left mouse button on the signal name.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

How To Highlight A Net In Schematic Window

1. To enable the highlight net function, click the **Highlight Net** icon.
2. Drag and drop any signal from either the **Source Code** window, or the **Signal Selector**, or the **Waveform** window, onto the schematic.

   If the **Highlight Net** icon is toggled on, the **Schematic** window highlights the selected signal in orange.
3. Click the up or down icon buttons, to move up or down to each connection of this net.

   You can move to any element in this highlighted network.

How To Locate a Schematic Element in a Source Code Element

Drag any element from the schematic to the **Source Code** window.

The **Source Code** window displays the source code for that element.

How To Display A Signal From the Schematic Window

Drag and drop any signal from the schematic into the **Waveform** window.

The **Waveform** window displays the waveform for the selected signal.

How To Find the Source Code From the Schematic Window

1. In the **Schematic** window, place the cursor in the center of the schematic graphical symbol.
2. Click the left mouse button.

   The Undertow Suite **Source Code** window displays the line or lines of code that created the selected schematic block.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

How To Start the State Window

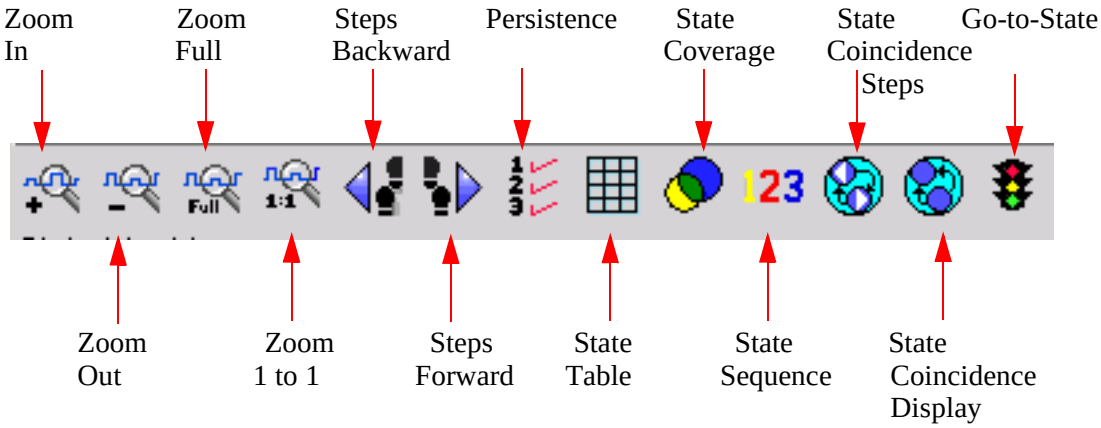
From the menus in the Source Code window, select one of the following three options:

Tools => ***State***, or

Windows => ***Tile Windows*** => ***State***, or

Windows => ***Tile Windows*** => ***State and Schematic***

FIGURE 135. Icons in State Window



How To Select a State Diagram

The state diagrams selector area is located at the top of the **State** window.

1. In this selector area, select the name of the state diagram to display.
2. To display more than one state diagram, use the following menu option to add another pane for each additional state diagram to display:

   Window => ***Add Panes (horizontal or vertical)***

   *Note: The selected state diagram opens in the pane that is currently in focus. If a pane is in focus, the **A** (located in upper left corner of a pane) has a yellow background.*

How To Step the State Diagram

To step the active pane that holds a state, click the **Step Forward** or **Step Backward** arrow icons on the **State Diagram** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

How To Find Source Code From the State Window

In the **State** window, use the middle mouse button to drag and drop a state bubble into the **Source Code** window.

The Undertow Suite **Source Code** window displays the line of code that generated the selected state bubble.

How To Show All Exercised States

To see exactly which states the simulator has exercised:

1. Click the **Persistence** icon.
2. Step the state diagrams from any time in the simulation, to any other time.

   The **State** window shows all states that the simulator exercised during the selected time period.

How To Show All Coincident States

1. Select any state in the active pane.
2. Click the **State Coincidence, Display** icon.

   The **Output Display** window lists any states, in all other state diagrams, that are coincident to this state.

How To Display a State Table

To display the state table for the active pane, click the **State Table** icon.

How to View Memory Contents

You can load the memory dumpfile from the **Waveform** window.

1. Select ***File*** **=>** ***Load Additional File*** from the menus in the **Waveform** viewer.

   The Undertow Suite adds another root in the **Chooser** section of the **Source** window.
2. Open the hierarchy and display the memory, in the same way that you display a signal.

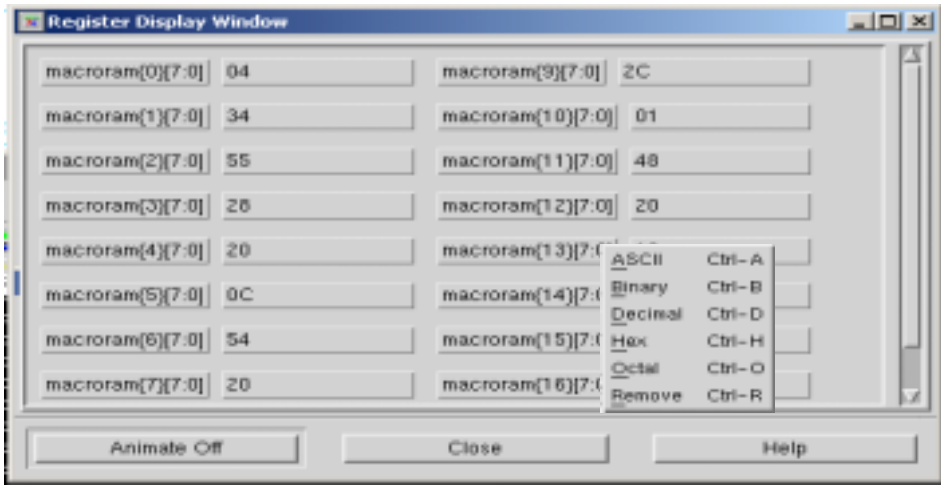
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The **Waveform** viewer displays the memory addresses, and the memory contents. User can also view the content through Register Display Window.

Register Display Window

Click on options->Register Window to open up the Register Disply Window

FIGURE 136. Register Display Window



User can click on any part in the waveform and Register Display Window will automatically update and display the content of the memory. User can also change the output format by left clicking on the register or:

1. ASCII (Ctrl-A)
2. Binary (Ctrl-B)
3. Decimal (Ctrl-D)
4. Hex (Ctrl-H)
5. Octal (Ctrl-O)

Or remove the Register Bit by clicking Remove (Ctrl-R)

How To Overlay Signals

1. To read in more than one signal file, from the menus in the Undertow **Waveform** viewer, select ***File => OpenAdditonalFile***.
2. Select ***Options => Properties => OpenFiles***.

3. Set the *base* file and the *overlay* file.

4. Use the **Options** icon in the **Choose** window (A/B/C with check marks) to toggle *overlay* on.

   Whenever you select and display a signal from the specified *base* file, Undertow Suite automatically overlays the signal from the specified *overlay* file.

How To configure the Xsun server on Solaris to use a default 24-bit TrueColor colormap

5. Edit the file /usr/dt/config/Xservers

6. Edit the last line in this file, which is :

   :0 Local local_uid@console root /usr/openwin/bin/Xsun :0 -nobanner

   Add the arguments "-dev /dev/fb defdepth 24 defclass TrueColor" to the end of this line (all on one line)

   :0 Local local_uid@console root /usr/openwin/bin/Xsun :0 -nobanner -dev /dev/fb defdepth 24 defclass TrueColor

   Now when the X windows starts on the machine, it will be using a 24-bit

   TrueColor default colormap.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Undertow Suite Overview

The Undertow Suite is a graphical source code debugging tool for Verilog-XL and Verilog clone simulators. It uses PLI/FLI routines that are linked into Verilog/VHDL simulators, to communicate with the simulator while the simulator is running.

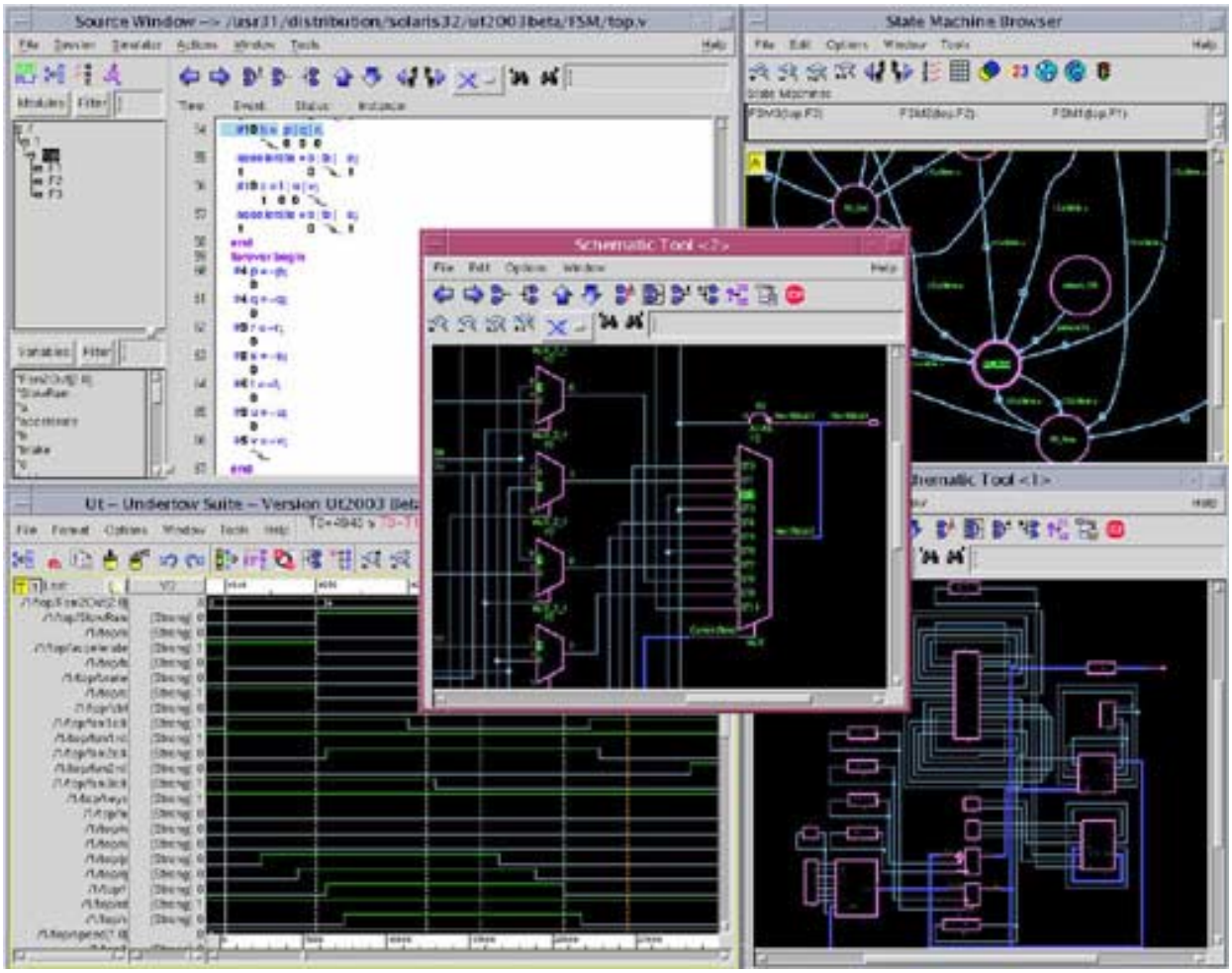


FIGURE 137. Examples of Undertow Suite Windows

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This section describes the basic operations of the Undertow Suite. You can use this set of tools to debug Source Code for Verilog or VHDL. In future versions, you will also be able to debug System C source code.

The Undertow Suite includes the following main windows:

Source Code Window

Use this window to view your source code. You can also use this window to interactively control the simulator, if the simulator is currently running. You can use the controls on the icon bar to:

- Step the simulator forward, in interactive mode, and
- Step the simulator backward (if you created a trace file), either in interactive mode or in batch mode, even if no simulator license is active.

Source Code Browser and Signal Selector

These windows are just to the left of the Source Code browser. You can use them to:

- Open the design hierarchy.
- Select a level of hierarchy
- Select signals to display in the waveform window.

In the upper window, you can toggle on the **Module** button, to display any module from the source code, in the Source Code window.

To find any signal in the Source Code window, you can selecting that signal in the Signal Selector, and toggle on the V**ariables** button.

Waveform Window

The **waveform** window, called *Undertow*, displays the waveform for the design. Waveform file formats can be:

- VCD.
- Fast Files.
- Files generated by the simulator, using PLI or FLI.

State Diagram Window

This window displays the state diagrams for the current design.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The State Diagram window includes a state diagram selection pane, at the top of the window.

- You can select the state diagram that you want to see.
- You can create a state diagram window with as many panes as you want, and load any state diagram into any pane.
- You can use icon controls to step forward, and step backward, any state diagram that is in focus.
- You can use several state diagram *analysis* tools, to analyze your state diagram:
  - **Persistence** shows which states you have entered.
  - **State Table** shows whether you have a complete state diagram.

  At any selected state that is in focus in the state diagram, **State Coincidence** shows the state sequences that the other state machines are going through.

Schematic Window

The schematic window displays the RTL, GATE, or RTL/Gate schematic of your design. The schematic displayed starts at the level that you defined.

- To display only a logic path, traced back from an output pin, double-click on the output pin.
- To display the entire schematic at this level, press the Schematic window icon, **Show Entire Schematic**.

The schematic window is annotated with the signal values, at cursor time 0, **Cursor Zero**, for every signal shown on the schematic.

When you display the entire schematic, you can graphically show where a signal is connected into the rest of your design:

1. Press the **Show Net** icon on the gate browser window.
2. Drag and drop a signal name from the **Waveform** window, **Source Code** window, or **signal selector**.
   - The schematic window highlights this signal in *orange*, so it is clearly visible, even in a very dense schematic drawing.
   - All drivers and loads connected to this signal are shown attached to this highlighted net.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- You can drag and drop any signal name to the waveform window, to see the waveform for this signal.
- You can also drag-and-drop any signal name to the source code window, to see where this signal is located in the source code.

All of these windows are synchronized in time, when you use the Undertow Suite.

All of these widows have drag-and-drop capability:

- You can effortlessly move data between the different windows.
- You can also correlate the view in one window with the view in any other window.

Basic Operation of Undertow Suite

For a new user, the easiest way to learn Undertow Suite quickly, is to read this description of the basic operation of the tool, and then run the *Quick Start Guide* for **Verilog**, **VHDL**, or **HDLLint** operations.

Undertow Suite Basics

Starting Undertow Suite

You can start the Undertow Suite in 3 different ways:

- **Using a command line**, or
- **Using the GUI** to select the files and simulator types, or
- **Using the project file**.

Regardless of how you start the tool, you can save a project file at any time, for future use in restarting the Undertow Suite.

Using Command Line to start Undertow Suite

The following factors determine the types of commands that you enter on the command line:

- Whether you are running your simulator in batch mode, or interactive mode.
- Whether you are using a non-compiled or compiled simulator.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Batch Simulation with Command Line

After you run your simulator and produce the files that you need, you can use the Undertow Suite to run *batch simulation*. Batch simulation debugs your source code without tying up the simulator license.

For batch simulation, the *only* items that you must specify on the command line are the source code files to use, and some of the other files that you need. This command line has the following format.

ut -iv <list of design files> -sigfile <signal_file_name> -tracefile <trace_file_name>

OR

ut -iv -f <source code file> -sigfile <signal_file_name> -tracefile <trace_file_name>

You can enter a list of design files, or a single file that holds the names of other files. Your design files can be:

- Verilog files, or
- VHDL files, or
- a combination of Verilog and VHDL files.

For example, if your design uses files named:

- top.v
- fsm1.v
- fsm2.v
- fsm3.v

then list these design files.

However, if a file named **source.vc** lists all of these files, then you can specify the *<source code file>* as **-f source.vc**.

Other_file_types

-sigfile *<signal_file_name>*

Select one of the following signal file types:

- VCD file
- Fast file
- Optimized file from PLI

- Optimized file from FLI
- .wav file form Modeltech 5.2 or earlier
- .wlf file from Modeltech 5.3 or later

-tracefile <*trace_file_name*>

The trace file is a file that records the execution of each line of source code.

Several Verilog simulators can produce a trace file, including:

- VCS
- Verilog XL
- FinSim

You can use the trace file as a *Virtual Simulator* while you use the Undertow Suite.

The following are examples of command lines:

%ut -iv -f source.vc -sigfile fsm.sigs -tracefile fsm.trace

%ut -iv top.v fsm1.v fsm2.v fsm3.v fsm4.v -sigfile fsm.dump

%ut -iv top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl -sigfile fsm.wlf

Interactive Simulation with Command Line

Non-Compiled Simulator

To run Undertow Interactive Simulation with non-compiled simulator(Verilog XL , NCVerilog XL), the command line format is as follows:

%ut -iv <*command-line option*> <*simulator_name*> *-f* < *source code file*> -*sigfle* <*signal file name*> *-tracefile* <*tracefile name* >

Note: . . The <*source code file*> is a list of design files and iv.v. For more information on <*trace file name*> and <*signal file name*>, refer to “Batch Simulation with Command Line” on page 288

The Command line options are as follows:

Simulator	Command Line Option	Simulator Name
Cadence NC Verilog/ XL, NC Verilog	-ncxlmode	ncxlmode
Cadence Verilog XL	-xl	verilog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

NC Verilog XL, Example:

%ut -iv -ncxlmode ncxlmode -f source -sigfile fsm.sigs -tracefile fsm.trace

Verilog XL Example

%ut -iv -xl verilog -f source -sigfile fsm.sigs -tracefile fsm.trace

Compiled Simulators

To Run Undertow Interactive simulation with compiled simulators(NC Verilog, ModelTech, VCS and Finsim), the command is:

%ut -iv <command line option> <simulator_exec/ arguments> <other simulator_options> -sigfile <signal_file_name> - tracefile <trace_file_name> -ivsimcmp "-f < source code file>"

Note: . The <*source code file*> is a list of design files and iv.v. For more information on <trace file name> and <signal file name>, refer to “Batch Simulation with Command Line” on page 288

The Command line options are as follows:

Simulator	Command Line Option	Simulator Executable/ Arguments
Cadence NC Sim	-ncverilog	ncsim “-f ncsim.args”
Model Tech	-modeltech	vsim <top_level_module_name>
Finsim	-finsim	TOP.sim
Synopsys VCS	-vcs	simv

When you run the simulator, the Undertow Suite command line automatically assumes that these files are already compiled into an executable.

NC-Sim, Example:

%ut -iv -ncverilog ncsim “-f ncsim.args” -sigfile ut.sigs -ivsimcmp “-f source”

or

%ut -iv -ncverilog ncsim worklib.top:v -sigfile ut.sigs -ivsimcmp “-f source”

In this example, worklib.top:v is the design snapshot name defined in ncsim.args

ModelTech, Example

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

%ut -iv -modeltech vsim top -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

FimSim, Example:

%ut -iv -finsim TOP.sim -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

VCS, Example:

%ut -iv -vcs simv -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Using GUI to start Undertow Suite

This is one of the simplest ways to start Undertow Suite. This method uses the various file selectors to select the necessary files, and the GUI buttons to define the simulator type.

1. In your .cshrc file, set the path to include your **UT_ROOT_DIR**, which contains the **ut** executable.

   *Note: When you run interactive simulation, you can normally start the **Undertow Suite** in the same directory where you are running your Verilog or VHDL simulator.*

2. To start Undertow Suite, type the command:

   %ut -iv

The Undertow Suite opens in this order:

1. The Undertow **Waveform** window.
2. The **Source Code** Window.

   The **Source Code Browser** is on the left side of the Source Code Window.

Batch Simulation using GUI

1. To select your source code files in the Source Code Window, use this menu item:

   File* => *Open* => *Source Code (Import Design)

   The file selector opens.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select the directory that contains your source code files.
3. Select one source code file at a time, and click **Add File To List** for each selected file.
4. Repeat Step 3 to select all of the source code files in your design.
5. When all of your files are listed in the lower text window, click **Apply** to load your design.

You can edit this window, to add any command that the simulator requires, to interpret these files. For example if all of your source code files are in a file named **source.vc**:

1. Select **source.vc**.
2. Add this file to your list.
3. Add **-f** (followed by a single space) in front of the file name, source.vc

   Make sure a space is between the **-f** and the file name. This tells both Undertow Suite and the simulator that all of your files are listed in the **source.vc** file.
4. When you have selected the files that you want, click **Apply**.

   The files are loaded and parsed into a database, where the Undertow Suite can use them.

Next select your waveform file.

1. In the pull-down menus for the Undertow **waveform** window, select:

   File => Open/Close File

   The file selector opens.
2. Select the waveform file to use.

You can use the following file types for the batch operation in Undertow Suite:

- VCD files: Standard Verilog XL dump file.
- .fast files: Optimized File, created by the **$vtDump** PLI routine.
- .sig files: Highly compressed files created by the **$vtDumpvars** PLI routine.
- .wav files: ModelTech VHDL, pre-version 5.3.

  .wlf files: ModelTech VHDL dump file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Interactive Simulation using GUI

Undertow allows users to invoke Interactive Mode with Command Window:

In the Souce Code Window, select:

Session -> Source Code Files/Simulator Options

and the following window pops up

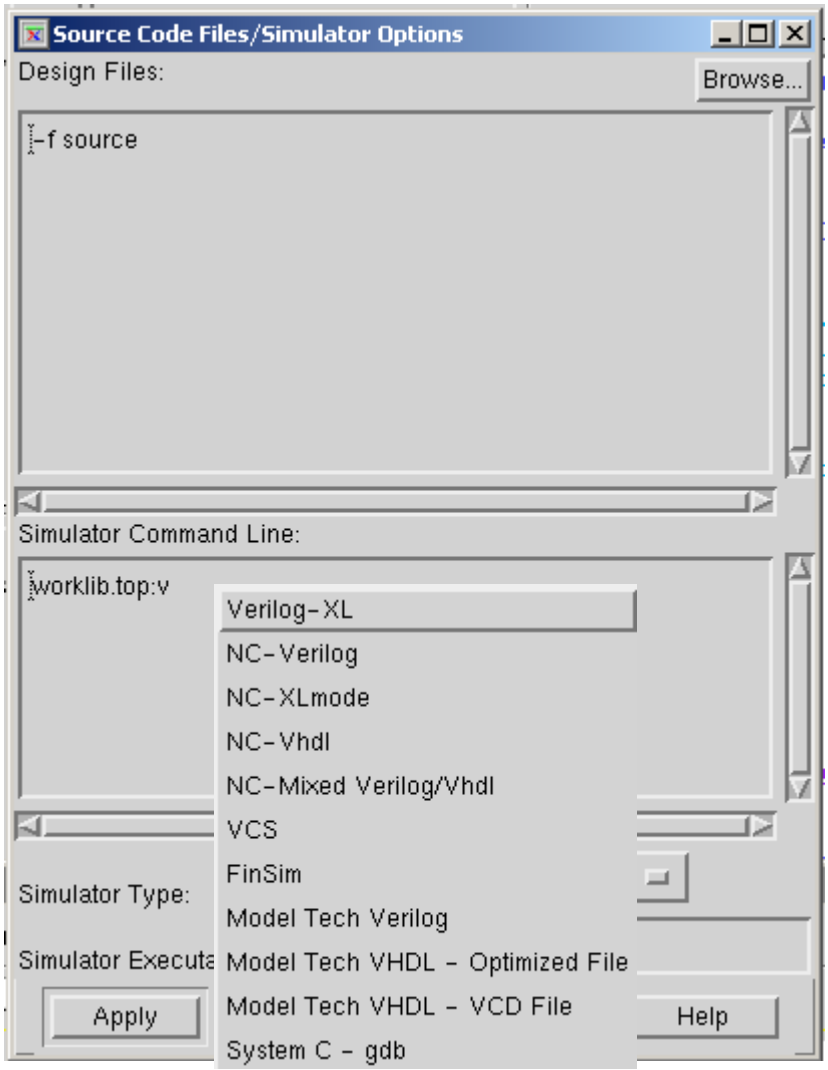


FIGURE 138. Source Code Files/Simulator Options Window

Simulator Type: pull down menu allows user to select the simulator type.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Simulator Executable:location of the simulator binary. Users need to specify the location of the simulator binary.

Compiled Simulator

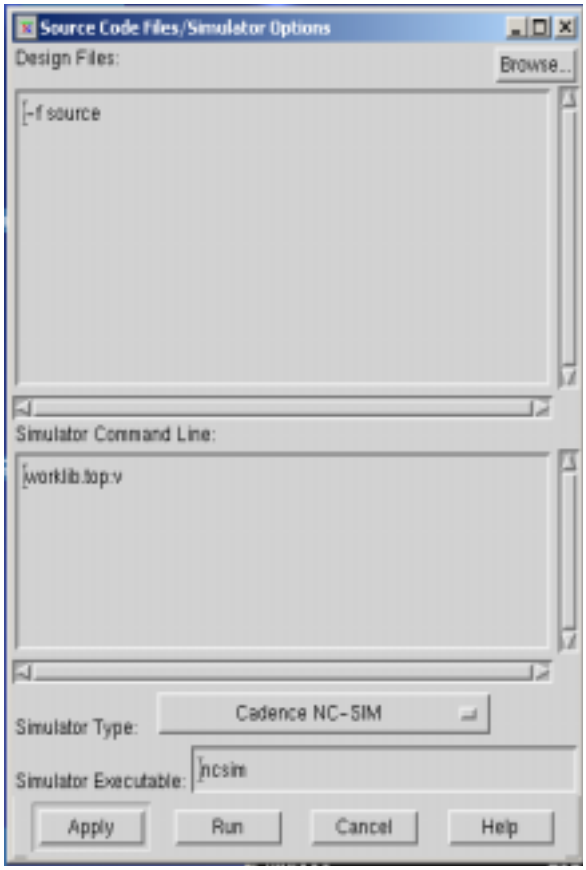
For Compiled simulators, the users need to specify the Design Files and Simulator Command:

Design Files: it cotains the source code file. In this example, the file source contains all the files that is necessary to run verilog simulations. This is equivalent to option -ivsimcmp “-f source” in the command line mode.

Simulator Command Line: This text field contains the command lines which will be passed into the simulator executable. This is equivalent to the simulator command in the command line.

ut -iv -ncverilog <*nc-excutable*> <*simulator_commands*> -ivsimcmp "-f source”

will appear as follows:



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Non Compiled Smulator

For non-compiled simulators such as Verilog XL or NCVerilog-XL, the window will be shown as follows:

Design Files/Simulator Command Line: is the command line argument that the users use to parse into the simulator.

ut -iv -xl verilog -f source <*optional simulator commands*>

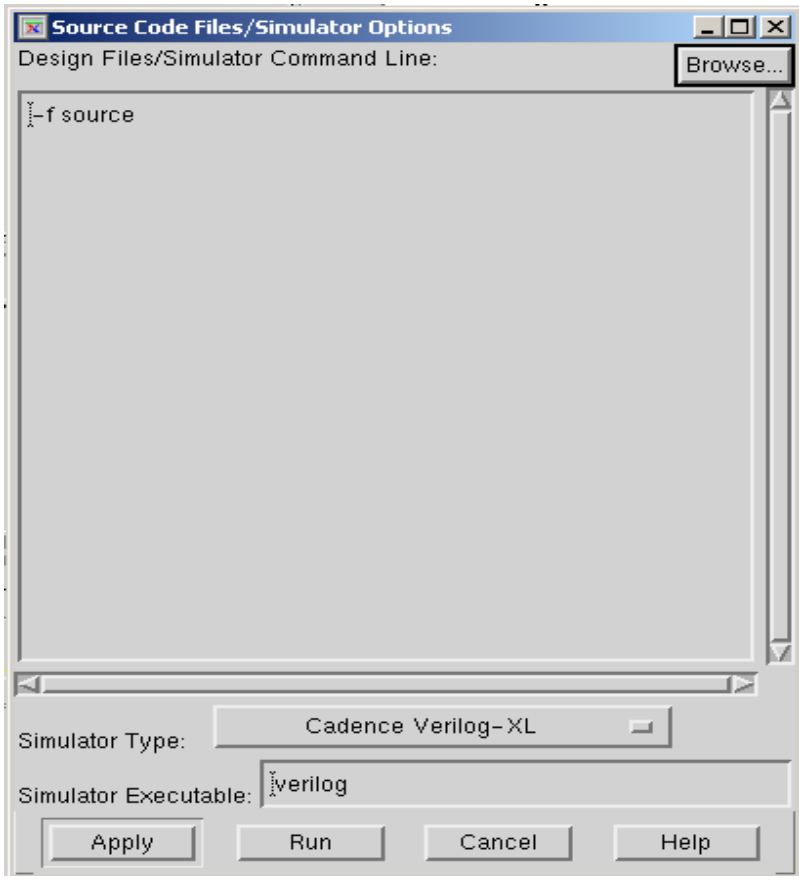


FIGURE 139. Source Code File/Simulator Options in Non Compiled mode

Running Interactive Simulations

After invoking Undertow suite with Interactive simulation mode, users can use run simulation interactively. However, there are certain points that user needs to make sure.

 Copyright Veritools 2004 All Rights Reserved

1. Load up the button File: Button file contains simulation commands that specific simulator has. User can select button files using ***open->File->Buttons***, select the correct button files.

Users are allowed to change the button file structure, please refers “Buttons Definition Option” on page 312

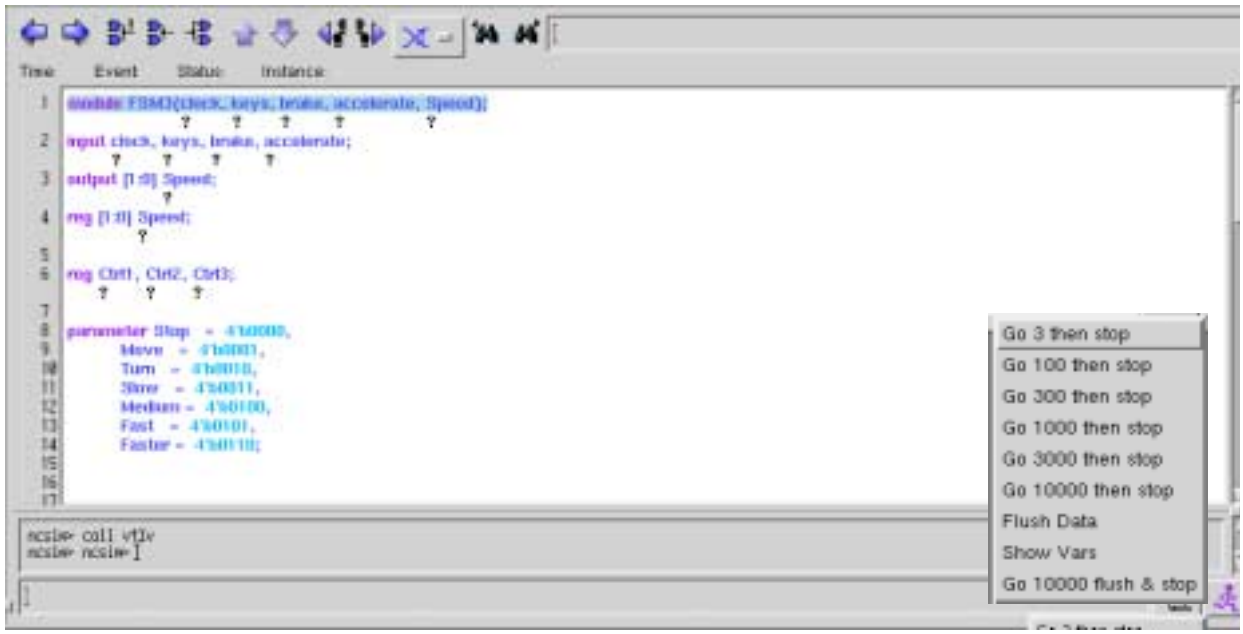
2. After the correct button file is selected, user can run simulation with:

Session -> Source Code Files -> RUN or

Simulator -> Run

The CLI window will appear as show:

FIGURE 140. Source code window with the CLI window



Clicking the Action Button on the right shows the button that you can run, user can also in their command with the command line at the bottom of the source code window.

Using a Project File to start Undertow Suite

Whenever you use the file selection method or the command line to start Undertow Suite, you can save a *project file*. To save a project:

Save->Project File

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The project file saves all of the information that Undertow Suite needs, so you can use the following command line to quickly start up your session and debug your source code:

%ut -iv -prj my.projectfile.prj

The project file includes the following fields.

This example shows a Verilog project, using the *Verilog-XL* simulator, which is a non-compiled (interpreted) simulator.

```
ivHierFile=
remoteHost=
verExecutable= /usr1/home/demo/ut8/ut8.2.8/verilog
executeDirectory =
dumpFileName= fsm.sigs
traceFileName = fsm.trace
connFileName = fsm.connectivity
hdllintFileName= FSM/fsm.lnt
vctReportFileName= vct.report
enableToggle= False
toggleFileName= vt.tgl
runSimImm = False
enableVct = False
dumpTraceInfo= True
createConnectivity= True
enableMonitorAll= True
remoteUsesOn= False
vcs = False
finsim= False
frontline= False
verilogxl= True
ncverilog= False
modeltech= False
modeltechVhdlOpt= False
modeltechVhdlVcd= False
enablePower= False
warnNoFanout= True
fastModeFlush = 60
simCmdLine= -s
simCmdLine= FSM/top.v
simCmdLine= FSM/fsm1.v
simCmdLine= FSM/fsm2.v
simCmdLine= FSM/fsm3.v
```

For a compiled VHDL simulator, the project file looks like the following:

```
ivHierFile=
remoteHost=
verExecutable= /usr1/home/demo/modeltech/5.4/modeltech/bin/vsim
executeDirectory=
dumpFileName= fsmvhdl.sigs
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
traceFileName= fsmvhdl.trace
connFileName= fsmvhdl.connectivity
enableMonitorAll= False
modeltech= False
modeltechVhdl= True
fastModeFlush= 60
simCmdLine= top
simCompileLine= top.vhdl
simCompileLine = fsm1.vhdl
simCompileLine= fsm2.vhdl
simCompileLine= fsm3.vhdl
```

The main difference between these examples is that, for a compiled simulator, the files that the simulator compiles are listed in the **simCompileLine** fields, and not in the **simCmdLine**.

Note: To use a project file when you start the Undertow Suite, use the standard project file that Veritools provides, and modify this file for your application.

Set up the following fields in your project file

- **verExecutable**: path and name of your simulator.
- **dumpFileName**: name of your waveform file.
- **traceFileName**: name of your trace file, if your use the *Virtual Simulation* feature.
- **hdllintFileName**: name of your *HDLLint* report.
- **dumpTraceInfo**: writes to a trace file, for the Verilog-XL simulator.

If you do not start your waveform file from the *top* of the hierarchy that you loaded from the source code, then you must map this waveform file hierarchy to the source code hierarchy.

If the hierarchies are different for the signal file and the source code file, you can also add a map command to map the two hierarchies:

%ut -iv -f top.vc -sigfile (*path*)/*signal_file_name*
-tracefile (*path*)/*trace_file_name* -map (*path*)/*map_file*

where ***map_file*** is the name of a file that contains the mapping command.

A map file is a simple ASCII text file, which contains one hierarchical mapping per line. Each line has the following format:

source.hierarchical.name = dump.file.hierarchical.name

Loading Files

To view waveforms:

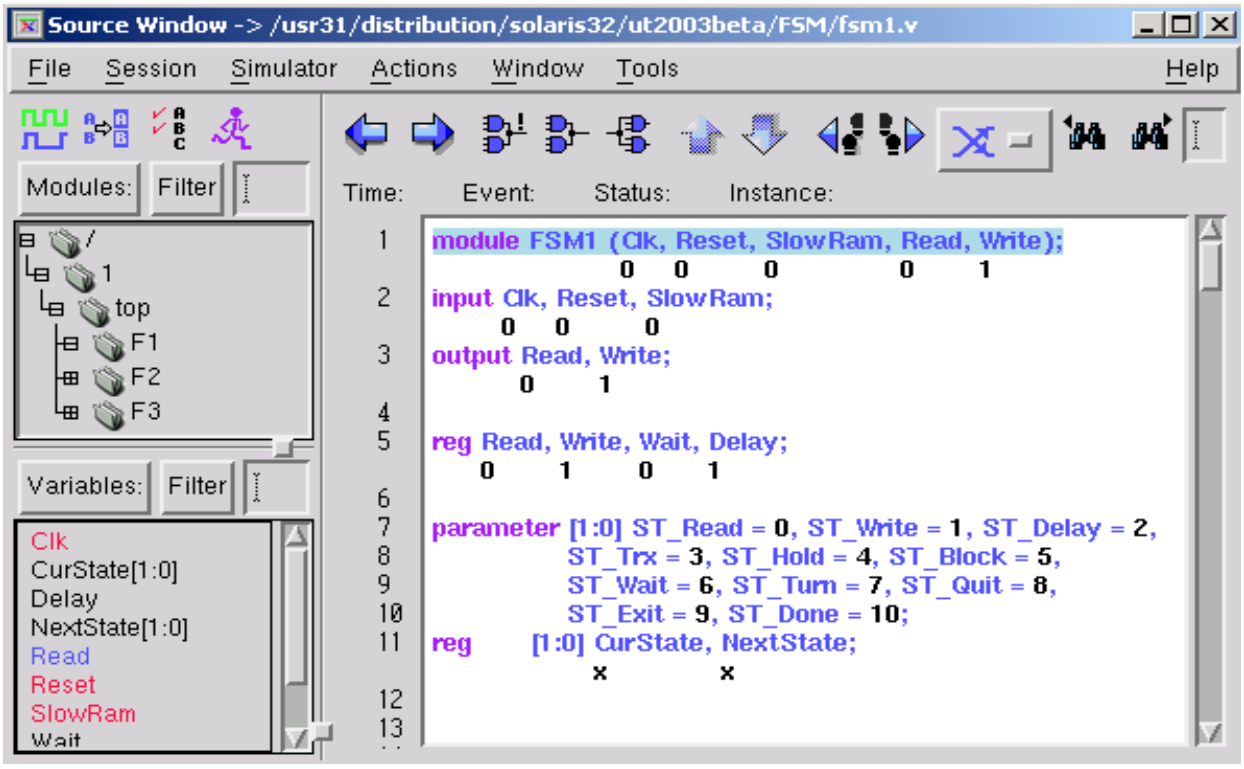
1. Open the source code browser window
2. Press the left mouse button on the upper-most + sign.

   This opens the hierarchy, down one level.
3. When signals appear in the **signal selector** window, select the module name at this level.

The module name turns black, indicating that you selected this level.

If Undertow found signals for the selected module, they appear in the **signal selector** window (the window below the **source code browser** window).

Source Code Window



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Source Code Window Icons

To invoke the **Source Window** from the icon bar in the trace window:

1. Place the mouse over the name of a module in the Undertow **choose** window.
2. Simultaneously press the right mouse button, and hold down the shift key.

   The **Source browser** opens. The code for the selected module appears straight in front of you, in the popped-up, color-coded **source** window.

*Note: You can also invoke the **Source Window** from the **Gate Browser**.*

Display— Displays the selected (highlighted) signals in the **waveform** viewer.

All—Selects (highlights) all signals in the **signal browser**.

Options—Shows a popup menu containing the following:
All—Finds the selected signal in all files.
Overlay Files—Lets you view multiple files on top of each other.
Hierarchy—Activates or disables the hierarchical module chooser. The hierarchical module chooser can save you a lot of mouse clicks.

- Modules appear as buttons.
- Each module button includes a triangle, indicating whether the module is expanded (triangle points down) or collapsed (triangle points to the right).
- Clicking on the triangle expands or collapses the module.
- Clicking on the module name displays the module's signals, in the **signal name list**.

Show Filenames—Shows the file name, and the hierarchy location.
Inputs—Shows only the inputs, in the selection window.
Outputs—Shows only the outputs, in the selection window.
Split Sweeps—Splits apart Sweeps.

Actions—Shows a popup menu that contains the following:
Copy—Copies the current selection to the buffer (for use with Paste).

*Note: The (Sun) **Copy** key has no effect in this window.*

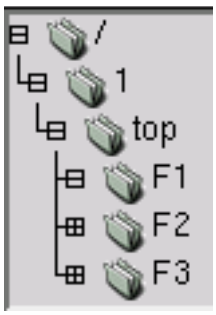
Text—Manipulate groups of signals, within a text window.
Explode Bus—Separately displays each bit in the selected bus.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Find Signals—Opens a box that you can use to find signals within a design.

1. Open the window.
2. In the **Variable name regular expression** box, enter the name of the signal that you want to find.
3. In the present module, or one of its submodules, select the toggle button that meets your needs.
4. Click **Begin Search**.

   The **Signals Found** window displays the signals that the search finds.
5. To highlight and display the signals, use the **Display** button.



Hierarchy Location Tab—Shows the current position in the hierarchy.

Modules:—If you toggle this button on, and left-click on a module in the hierarchy window, the source code for that module appears in the **Source Window**.

Filter—Lists only modules whose names contain the specified characters.

The following icons are located above the source code area.

- Click the left mouse button, to perform the action.
- Click the right mouse button, to display a help dialog that describes the button.

Back—Returns to the prior view, in the Source Code Window.

Forward—Shows you the prior back view.

Display Current Drivers—Shows the current drivers for your selected signal.

Show Drivers—If you toggle this icon on, and click on a variable, the drivers appear for the selected variable.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Show Loads—If you toggle this icon on, and click on a variable, the loads appear for the selected variable.

Up & Down—If **Show Drivers** or **Show Loads** finds more than one driver or load, the **Up** and **Down** arrows are enabled. The color of these arrows changes from gray to black.

Step Forward—Steps the **simulator** forward one event. If the simulator is not at the current time, or is not in interactive mode, then this icon steps the **source code window** ahead one event.

Step Backward—Steps the **simulator** backward one event. If the simulator is not at the current time, or is not in interactive mode, then this icon steps the **source code window** back one event.

Search Forward/Backward—Moves the cursor to the next or previous occurrence of the search signal item.

Search Signal—Sets the search item selector to one of the following:

- Transition
- Leading Edge
- Trailing Edge
- Value
- Strings

Use a text box in *conjunction* with both the search signal items value and the string. Enter the *value* or the *text string* to search for.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Source Code Menus

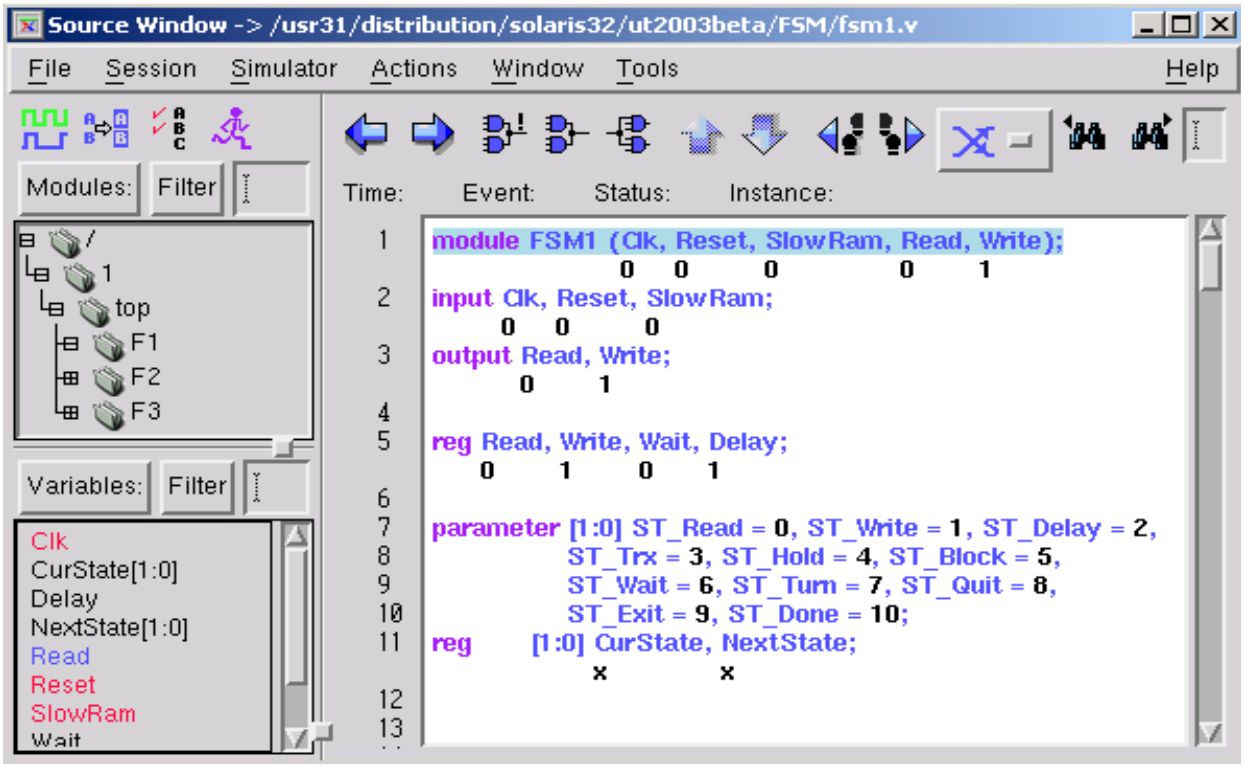


FIGURE 141. Source Code Menus

This section describes the six pull-down menus in the **source** window:

- File
- Session
- Simulation
- Actions
- Window
- Tools

 Copyright Veritools 2004 All Rights Reserved

File menu for the Main Undertow Suite Window

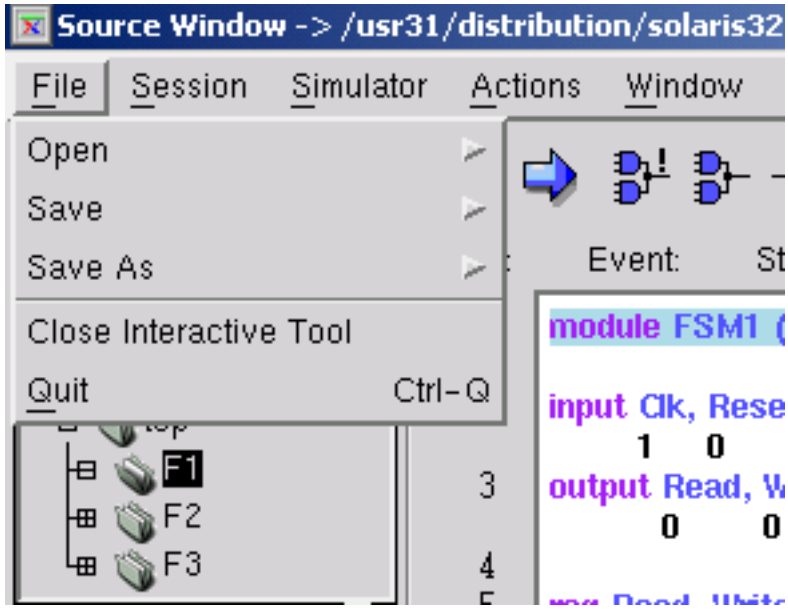


TABLE 48. File Menu

Menu Item	Description
Open	Opens the selected file (see Section , “File Open Menu”)
Save	Saves the current project.
Save As	Save to a new current project file.
Close Interactive Tool	Closes all windows except the **waveform viewer**.
Quit	Exits the Undertow Suite.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Open Menu

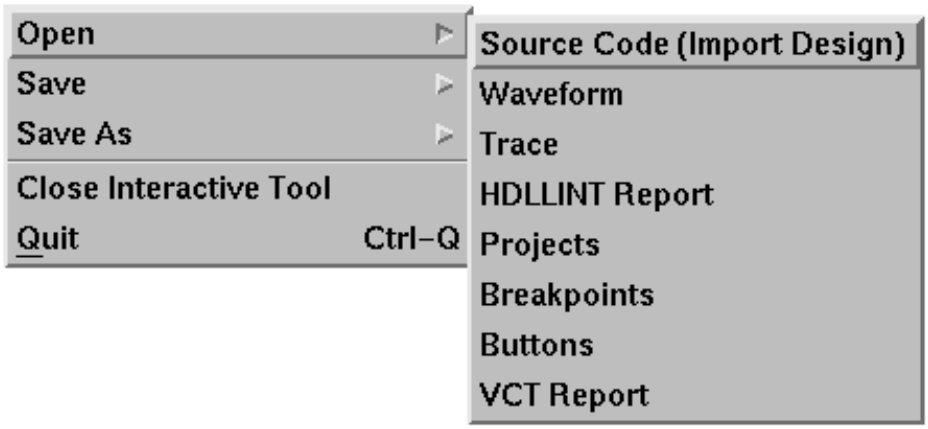


TABLE 49. File Open Menu

Menu Item	Description
Source Code (Import Design)	Displays a file dialog box for loading *source* files. .
Waveform	Displays a file dialog box for selecting a *signal* file.
Trace	Displays a file dialog for selecting a *trace* file.
Projects	Displays a file dialog for opening a project.
Breakpoints	Displays a file dialog for loading a *breakpoints* file.
Buttons	Displays a file dialog for opening a non-default *buttons* file.
VCT Report	Displays a file dialog for opening a *vcd report* file.

File => Open => Source Code (Import Design) Dialog

Use the source dialog to specify your Verilog or VHDL source files. You can add as many files as you want. To load a source file:

1. Use the **Directories** panel on the left to navigate to the files directory.
2. To select the file, highlight the file name in the **Files** panel, on the right side of the dialog.
3. When the file name is highlighted, click the **Add File To List** button in the dialog box.
4. Repeat step 3 for each source file that you want to open.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

5. When you finish selecting the source code files, click the **Apply** button in the dialog box.

   The module browser panel, in the **source code** window, displays the modules contained in these source files.

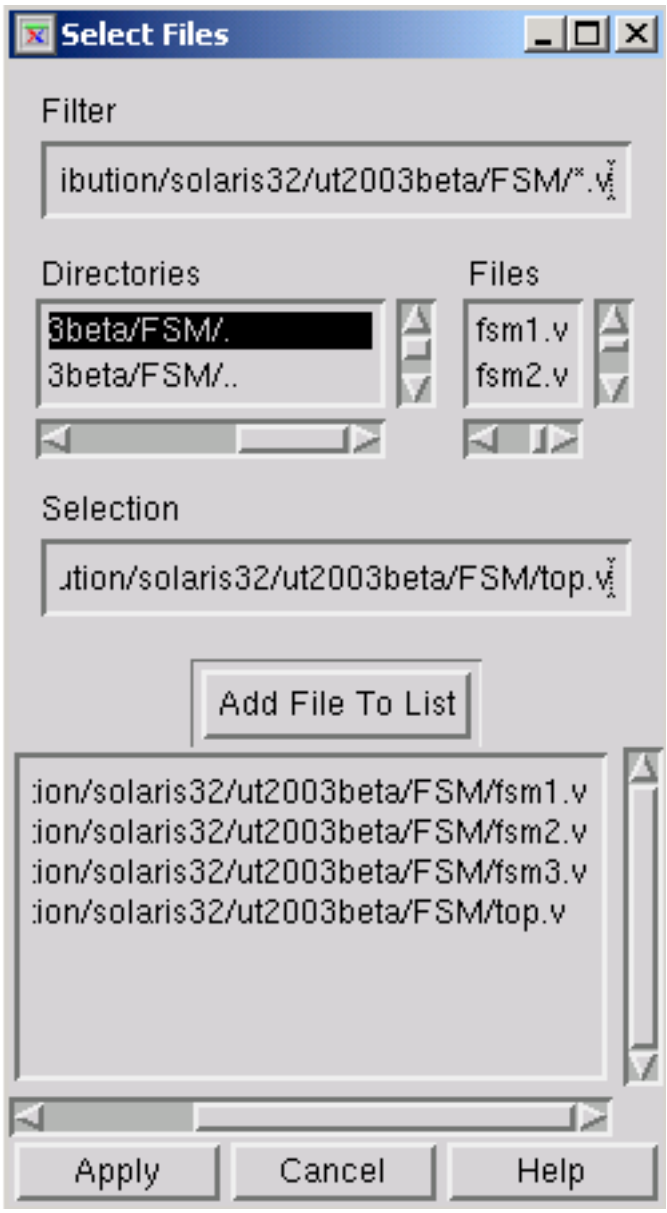


FIGURE 142. Open Source Dialog

You can add source files from the **Files** dialog box, shown in Figure 142, “Open Source Dialog”:

1. To select the file, highlight the file name in the **Files** panel (on the right side).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select the **Add File To List** button.

   The file name appears in the text area, in the bottom half of the dialog.

3. When you finish selecting the files to load, click **Apply**.

   The files are loaded into Undertow Suite, and are ready to view.

*Note: The **UT_WORK_DIR** environment variable sets the default directory that this dialog box opens.*

File Save Menu

The ***File*** **=>** ***Save*** submenu includes three items, described in Table 50, “File Save Menu”.

TABLE 50. File Save Menu

Item	Description
Projects	Saves any changes that you made to the *project* settings.
Breakpoints	Saves the current breakpoints to the existing breakpoint file. If no breakpoint file exists, a file dialog opens, to create a new one. To reload this file, use the ***File*** **=>** ***Open*** submenu, as described in Section , “File Open Menu”.
Buttons	Saves buttons.

Save As Menu

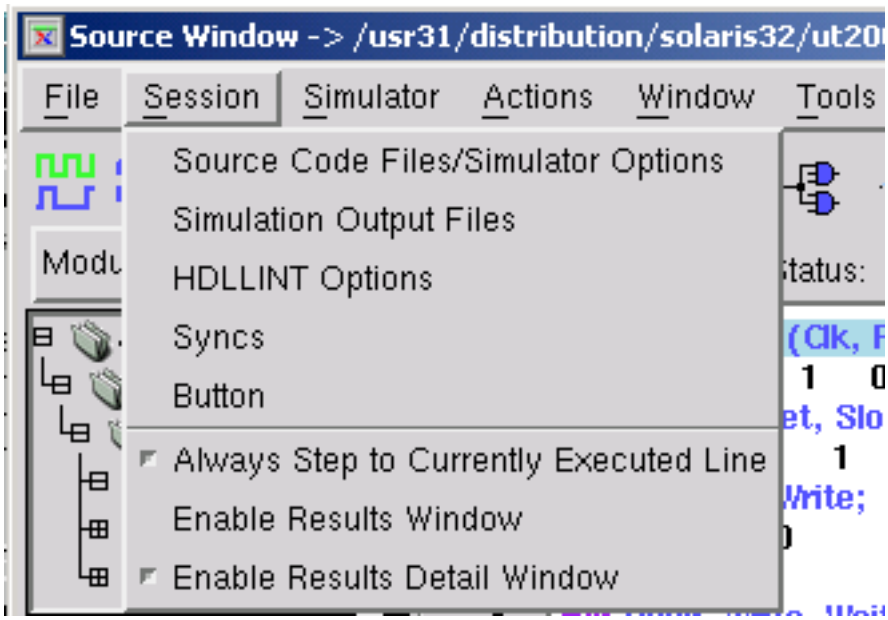
Similar to the ***File*** **=>** ***Save*** menu as describe above, but you can save files with different name.

Close Interactive Tool Menu Item

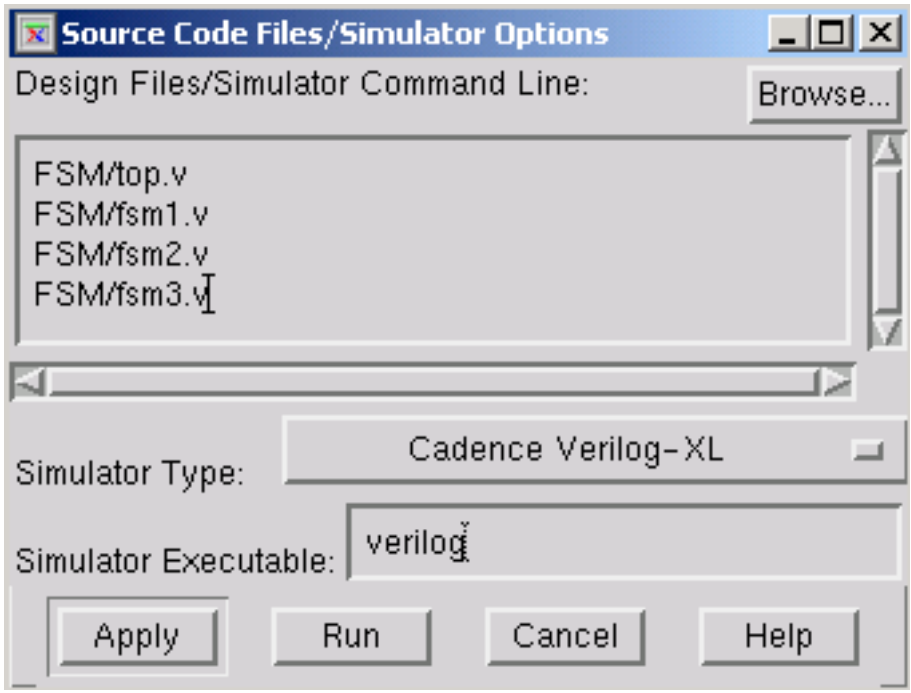
This menu item closes the **Source Code** window. You can continue your session using only the Undertow **Waveform Viewer**.

 Copyright Veritools 2004 All Rights Reserved

Description of the Sessions Menu



Source Code Files/Simulator Options

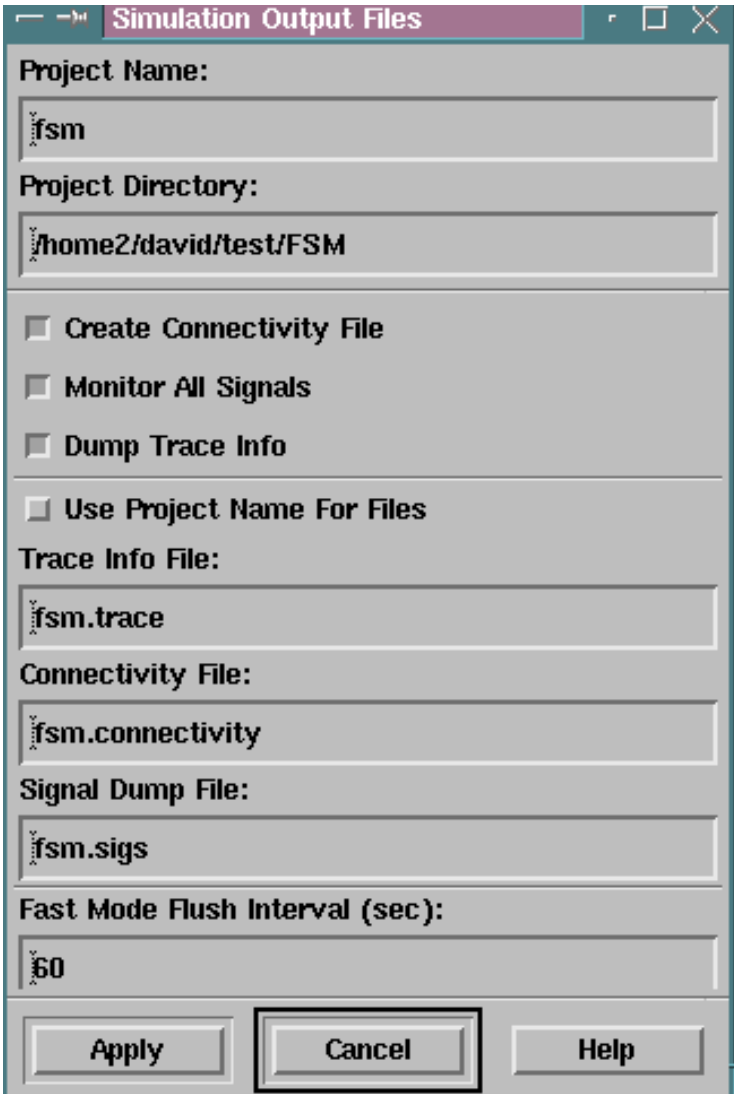


 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 51. Source Code File Simulator Options

Attribute	Description
Browse	Specifies the file names, and the commands to pass to the simulator
Simulator Type	List of supported simulator types.
Simulator Executable	Path to where the simulation program is installed.

Simulator Output Options



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 52. Simulator Output Options

Field	Description
Project Name	Project file name.
Project Directory	Path to the project *work* directory.
Create Connectivity File	Creates a Connectivity file for the Schematic Browser.
Monitor All Signals	If you check this box (the box appears sunken), then the **Interactive Tool** monitors all state changes, for all signals in the simulation.
Dump Trace Info	Dumps all events.
Use Project Name For Files	Uses the name of the project when you create trace, dump, and connectivity files.
Trace Info File	This field indicates the name of the file to use for dumping *trace* information. This is also the name of the trace file that you load for a *Batch Run* simulation.
Connectivity File	This field indicates the name of the file to use when you load connectivity information, using **Batch Run**. You can use this option if you when to want to review simulation later.
Signal Dump File	This field indicates the name of the Signal Dump File, when you load files using **Load Batch Run**.
Fast Mode Flush Interval	This field indicates how often to dump data to a *Fast File*. The default is 20 seconds.

Sync Options

Use the **Sync Options** dialog to selectively enable or disable synchronizations. Undertow supports three types of synchronizations (see Table 53, “Sync Options”).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

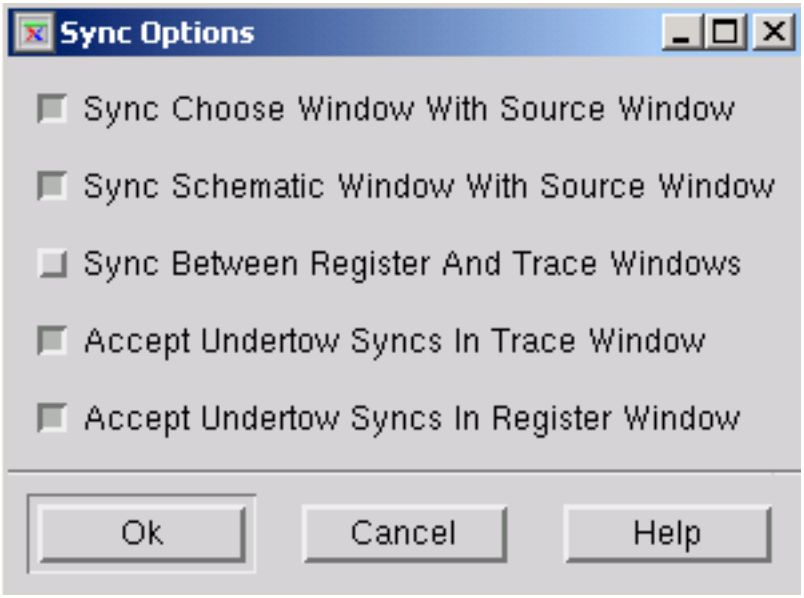


TABLE 53. Sync Options

Option	Description
Sync Choose Window With Source Window	This option synchronizes the **Choose** window with the **Source** window. When you select this option, the box appears to be pushed in.
Sync Between Register And Trace Windows	This option synchronizes the **Trace** Window and the **Register** Window. That is, both windows always display information about the *same* simulation time. When you select this option, the box appears to be pushed in.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 53. Sync Options (Continued)

Option	Description
Accept Undertow Syncs In Trace Window	This option synchronizes the **Trace** Window and the **Undertow** (waveform) window. If you start Undertow from the Undertow Suite, then whenever you move the main cursor, Undertow sends sync requests to the Undertow Suite. If you set this option, then the **Trace** Window honors these sync requests, and moves to the appropriate simulation time. When you select this option, the box appears to be pushed in.
Accept Undertow Syncs in Register Window	This option synchronizes the **Register** Window with the **Undertow** (waveform) window. If you start Undertow from the Undertow Suite, then whenever you move the main cursor, Undertow sends sync requests to the Undertow Suite. If you set this option, then the **Register** Window honors these sync requests, and moves to the appropriate simulation time. When you select this option, the box appears to be pushed in.

Buttons Definition Option

Use this option to modify and create new buttons.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

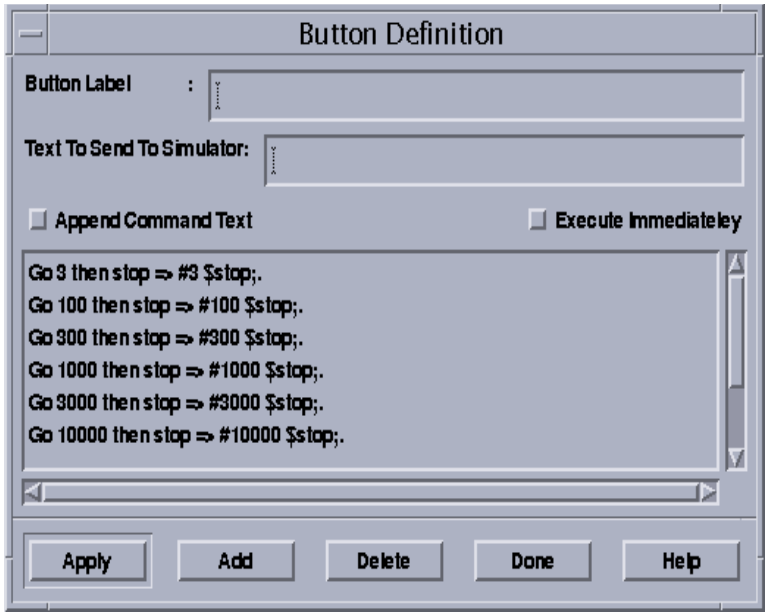


TABLE 54. Definitions of Buttons

Feature	Description
Button Label	This text field defines the label for a selected button.
Text To Send To Simulator	This text field specifies the command text that the Undertow Suite passes to a Verilog interactive prompt.
Append Command Text	This field appends the command text to any text that is already in the command input area. If you do not select this option, then the command text replaces any existing text in the command input area. You can use this option, and push multiple buttons, to create a command. For example, a button might send a signal name as an argument to several other button commands.
Execute Immediately	This field adds a carriage return at the end of the command text, so that this button immediately executes the specified command in the Undertow Suite.

The bottom of the dialog contains a list window, containing the names of all currently-defined buttons.

1. Use the mouse to select a button from the window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The definition of the button appears in the upper section of the dialog.

2. Edit the definition, and click on one of the following:

   - **Add** to create a new button, or
   - **Apply** to modify the definition of the selected button, or
   - **Delete** to delete the currently-selected button.

You can save button definitions to a file, and restore them from a file. To load button files, from the main menu bar, select ***File*** **=>** ***Buttons***.

Always Step To Currently-Executed Line

This option steps a **Source Code** Window to the last line that the simulator executed.

- In **Interactive** mode, the simulator executes *one* line of code. The Source Code Window moves to the next event in the current file, only after the simulator executes that line of code.
- If you toggle this option *off*, the **Source Code** Window displays the next event in the current file.

The **Source Code** Window opens a dialog box that states "searching forward until the next event on the current file is reached."

Enable Results Window

This option enables the **results** window to open. This window opens during a **Find assignments** operation if either:

- **Show Driver** finds more than one driver, or
- **Show Loads** finds more than one load.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Simulator Menu

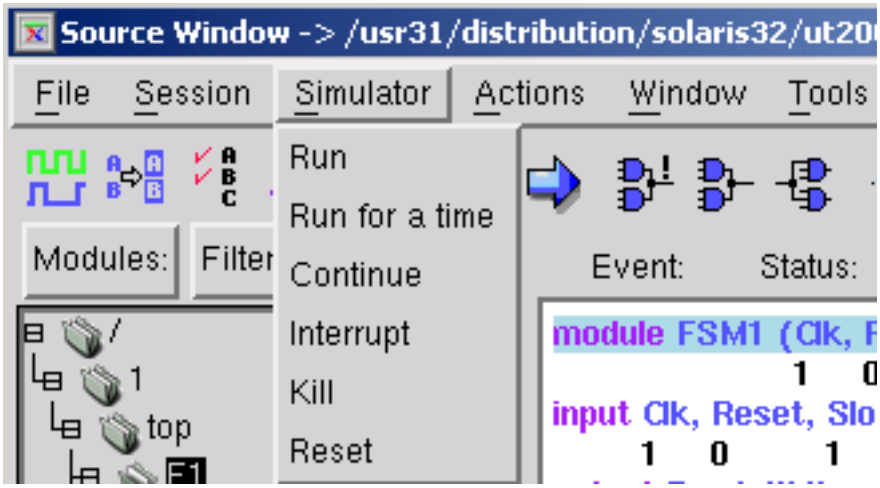
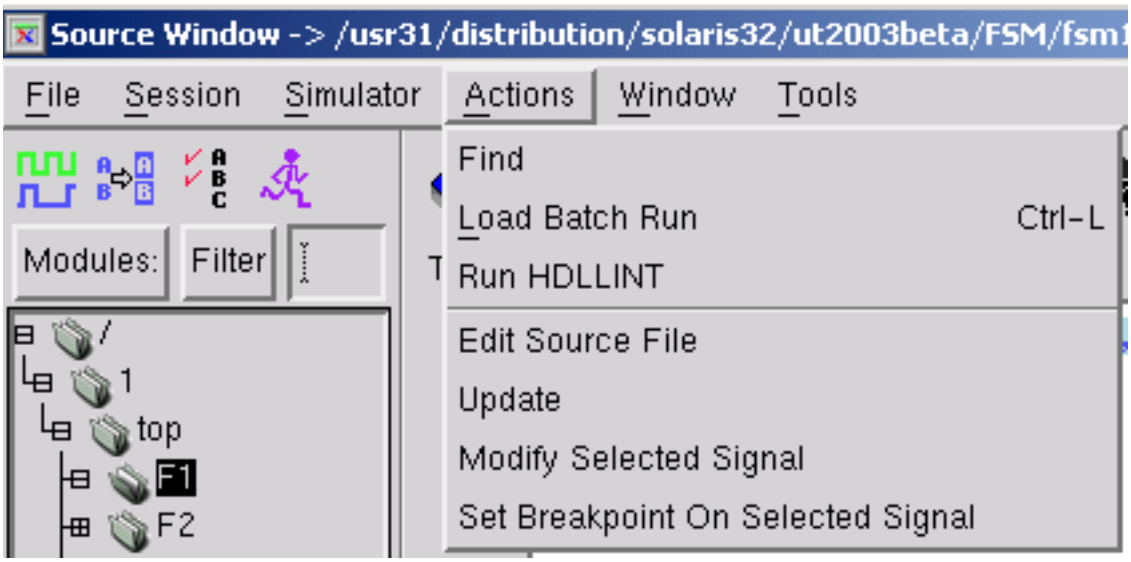


TABLE 55. Simulator Menu Options

Item	Description
Run	This option starts the simulator. If you entered the correct information into your project file, the **Source** Window and the **Undertow** waveform window open, and the **Control** Window shows the `Verilog` prompt.
Run for a Time	This options is same as Run option with a specific time.
Continue	This option resumes a simulation, after an interrupt.
Interrupt	This option interrupts the current simulation.
Kill	This option stops the current simulation. This is equivalent to **control**.
Reset	Use this option to set the simulator to time *zero*. A confirmation box appears.
History	This option shows the simulation history.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Actions Menu



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 56. Action Menu

Menu Item	Description
Find	
Load Batch Run	This option loads the following: • The data that you specified in the **Monitors** and **Dumps** window. • The trace file, if specified. • The signal file. If you click **Load batch Run**, the Undertow Suite also parses the following: • Verilog commands. • Connectivity file. • Stated machine file from the parsed source code. Undertow Suite then saves these files into the project directory. • The **project** file defines the name of the **connectivity** file, usually (*project_name*).connectivity. • The state machine name is (*project name*).dot. • When loaded into the State Diagram window, Undertow Suite creates a new file, (*project_name*).dot.done.
Run HDLLINT	This option runs the **HDLLint** tool on the files that the *project* file specifies, and automatically loads the results into the **Source Code** Window. • Use the **GoTo** icon to find the results. • Use the **Detail** window (the **Show Details** icon) to view the details of the *HDLLint* results.
Edit Source File	Use the **EDITOR** environment variable to select either the *vi* or *emacs* text editor. For example: `%setenv EDITOR vi` When you select this option, the selected text editor opens the file that appears in the **source** window.
Update	This option updates the **Source Code** Window, to the current line and file where the simulator is located.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 56. Action Menu (Continued)

Menu Item	Description
Modify Selected Signal	After you select a signal from the **Source Code** Window, you can click this button. For details about selecting signals, see the description above. When you click this button, the **Alter Signal** Window pops up, the name of the selected signal appears in the signal text field of the **Alter Signal** Window. You can use the **Alter Signal** Window to change the current value of a signal.
Set Breakpoint On Selected Signal	This option stops the simulation at the selected signal.

Window Menu

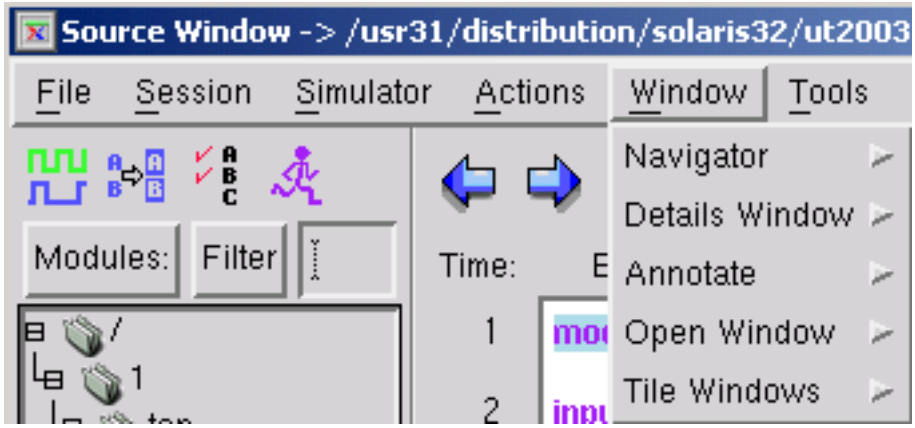


TABLE 57. Window Menu Options

Menu Item	Description
Navigator	Moves either the Source Window or the Waveform Window to the top.
Details Window	• Select **Simulator** to open a window on the bottom of the Source window, where you can enter simulator commands. • Select **Results** to open a window that reports simulation messages.
Annotate	Selects whether to show the bus values in the Source window, in Binary or He xi decimal format.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 57. Window Menu Options (Continued)

Menu Item	Description
Open Window	Displays a pull-down menu. When you select a window type from this menu, one of the following windows pops up during interactive debugging mode: • Results • Trace Filter • Break Point • Search • Alter Signal • Inspect Signal
Tile Window	This option opens a pull-down menu for *tiling* various windows. Tiling fits windows to your screen, and lets you fully view multiple selected windows side-by-side on your screen.

Window => Nevigator => BreakPoints Menu Item

This menu item is visible *only* if you have either opened a breakpoint file , or opened the breakpoints dialog box (***Windows*** **=>** ***Open Window*** **=>** ***breakpoints*** menu item You can use the breakpoints window to:

- add breakpoints
- delete breakpoints
- modify breakpoints
- save breakpoints
- load breakpoints

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

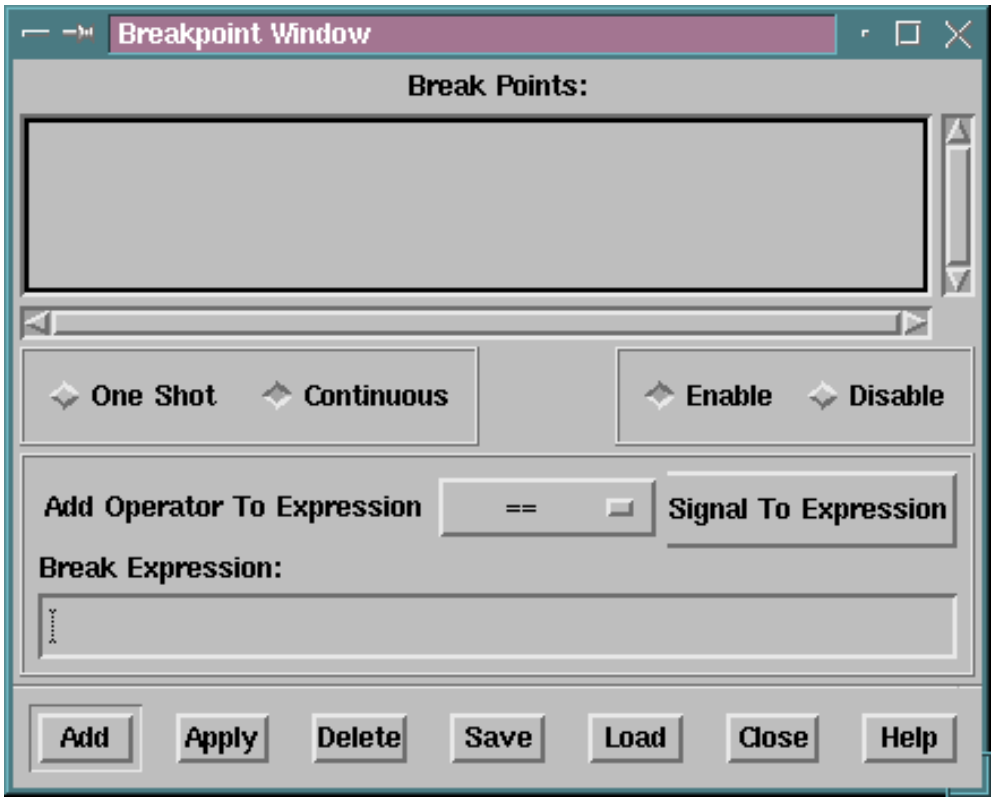


FIGURE 143. Break Points Dialog

The top section of the **Breakpoint** Window lists all of the currently-defined breakpoints.

- To set a new break point, click the left mouse button on the line number of the source code.

  A red dot displays to the left of the line number, for each breakpoint that you set.

- To turn off a particular breakpoint, left-click on its red dot.

  The dot turns green, showing that the break point is not active.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

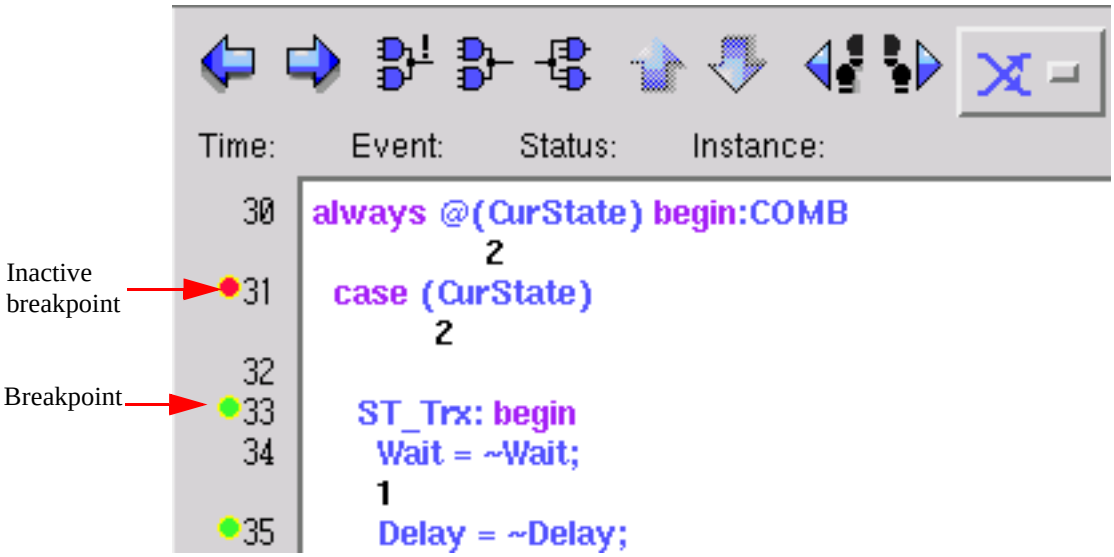


FIGURE 144. Setting Breakpoints

TABLE 58. Break Points Dialog box

Item	Description
Break Points	Lists all of the currently-defined breakpoints
One Shot	Automatically disables a breakpoint after it triggers.
Continuous	Leaves a breakpoint enabled (persistent) after it triggers.
Enable	Turns on a breakpoint.
Disable	Turns off a breakpoint.
Add Operator to Expression	1. Press and hold the left mouse button on the button next to this label. A list of all supported operators pops up. 2. Select an operator. The selected operator appears in the text field, below the Break Expression label. *Note: Another way to add an operator is to select the text field below the Break Expression label, and use the keyboard to enter the expression.*

 Copyright Veritools 2004 All Rights Reserved

TABLE 58. Break Points Dialog box (Continued)

Item	Description
Signal To Expression	1. Click the **Signal To Expression** button. The Signal Choose Window pops up. 2. Use this window to browse the design hierarchy, and to select a signal to add to the expression. The selected signal name appears in the **Break Expression** text field. 3. Edit this field directly, to modify the signal name, so it is a valid Verilog bit select or part select.
Break Expression	This text field contains the break expression. You can edit this field, to define a complex break expression.
Add	This button creates a new breakpoint, defined by the current contents of the dialog. The Undertow Suite parses the break expression, and sends it to Verilog PLI routines, which detect the breakpoint.
Apply	Use this button to modify the definition of the currently-selected breakpoint. The new definition is the current contents of the dialog. The Undertow Suite sends the changes to Verilog PLI routines, which detect the changed breakpoint.
Delete	This button deletes the currently-selected breakpoint. The Undertow Suite notifies the Verilog PLI routines, which end detection of the breakpoint.
Save	This button pops up a standard file dialog. Use this dialog to select the path and name of a file, in which you save the definitions of all current breakpoints.
Load	This button pops up a standard file dialog. Use this dialog to select a previously-saved breakpoint file. The Undertow Suite loads and parses the new breakpoints. It then sends them to the Verilog PLI routines, which immediately detect the newly-loaded breakpoints.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Window => Navigator => Source Window

This option pops up the **source code** window.

Window => Navigator => Wave-Display

This option expands the waveform view (if you reduced it to an icon), or pops the waveform view to the top (if other windows obscure it).

Windows => Details Submenu

Details windows appear at the bottom of the **source code viewer**. The following sub-sections describe these windows.

Simulator Menu Option

When toggled on, the **Simulator** menu option adds two text windows to the bottom of the **source code** window.

- Use the lower text window to enter commands from the keyboard, instead of using the file pull-downs and dialog boxes.
- The upper text window displays the generated output messages for the commands that you entered.

Annotate Hexadecimal/Binary

This option selects whether to show bus values as Hexadecimal or Binary.

Open Window

This menu item lists several windows, which the following sub-sections describe.

Results

This dialog box displays the results of any one of the following:

- Show Drivers.
- Show Loads.
- Display assignments search.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

You normally enable this window from the ***Session*** => ***Enable Results Window*** pull-down menu item.

Trace Filter

This option pops up the Trace Filter dialog box, as shown in Figure 145, “Trace Filter Dialog”. Use this dialog to specify one or more modules, from which you enable the trace.

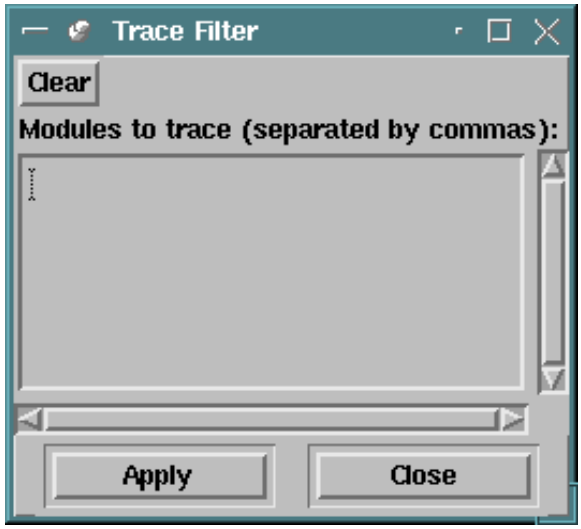


FIGURE 145. Trace Filter Dialog

Break Point

Opens the Breakpoint dialog box. See Section , “Window => Nevigator => BreakPoints Menu Item” for more details about this dialog.

Search

This option opens the Search Dialog Window.

1. Use this dialog to search for a variable (signal).

   The search results show the file and line number where the signal was found.
2. Double-click on this file and line number.
3. This line appears in the **source code** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

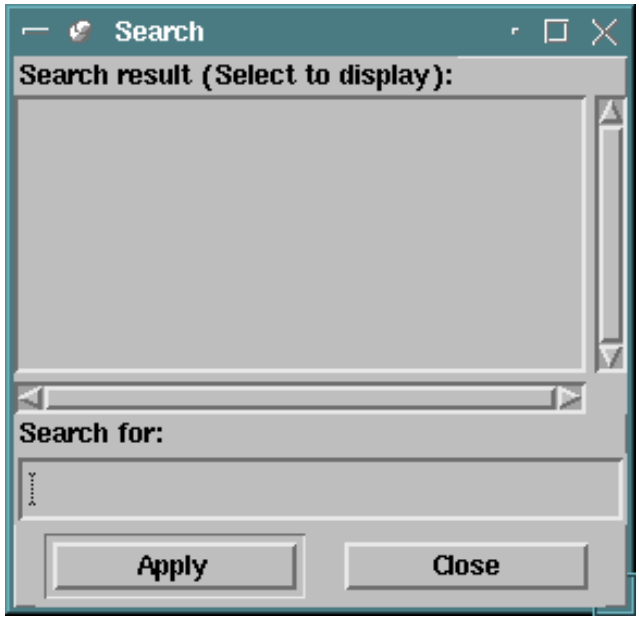


FIGURE 146. Search Dialog Box

Alter Signal

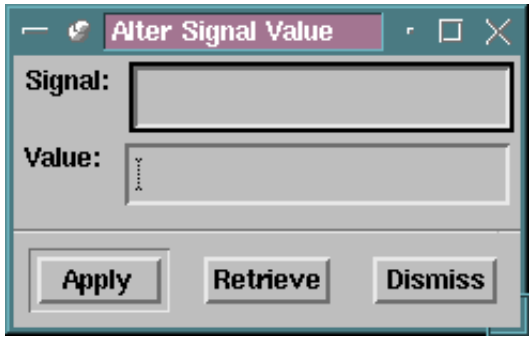


FIGURE 147. Alter Signal Dialog

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Inspect Signal

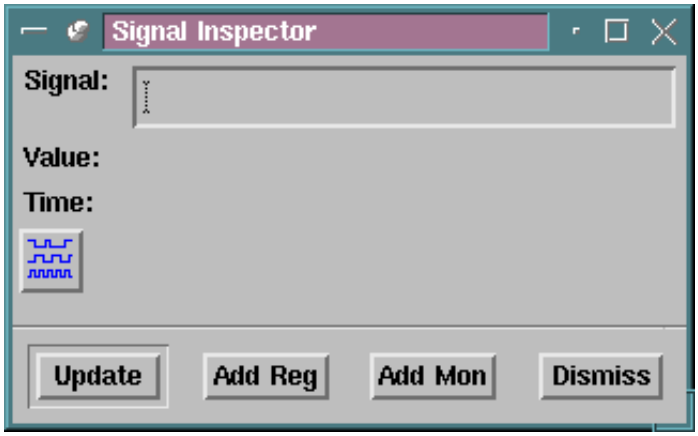


FIGURE 148. Inspect Signal Dialog

Register

This window allows the user to view the value of signals during simulation. The window contains a label at the top which shows the last simulation time for which signal values were collected. Below this is a time line. The user can move the thumb dial of the time line to see the state of signals at various different points during the simulation. Below the time line is a scrollable window which will contain two fields for every signal. The left-most field will contain the signals complete hierarchical name. The right-most field will contain the signals state for the time specified by the time line. The user can press either of these fields to pop-up a format menu. The format menu allows the user to change the displayed signals value format. Possible formats are ASCII, Binary, Decimal, Hex and Octal. The format specified will only affect the signal which was selected, thus signals may be displayed in different formats. At the bottom of the window is a Choose button. If this is pressed a hierarchical choose window will pop-up. The user can move up and down the hierarchy choosing signals to be added to the Register Display Window. The Close button will pop-down the window, but the data will remain and will still be collected as simulation progresses. To bring the window back up, select the Open Register Display Window from the Actions menu item in the main menu bar.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Tile Window

This option positions (tiles) various windows associated with the Undertow Suite, including:

- Undertow
- waveform viewer
- Source Code Browser

The tiles feature automatically sets and displays windows to a optimal size for viewing, to maximize window usage. Table 59, “Tile Window” describes the tile options.

TABLE 59. Tile Window

Menu Item	Description
All	Tiles all windows.
States	Tiles the **state** window with the **source code** and **waveform viewer** windows.
Schematics	Tiles the **schematic** window with the **source code** and **waveform viewer** windows.
States and Schematics	Tiles the **state** and **schematic** windows with the **source code** and **waveform viewer** windows.
HDLLINT	Tiles the **lint** window with the **source code** and **waveform viewer** windows.
HDDLINT and Schematics	Tiles the **lint** and **schematic** windows with the **source code** and **waveform viewer** windows.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Tools Menu

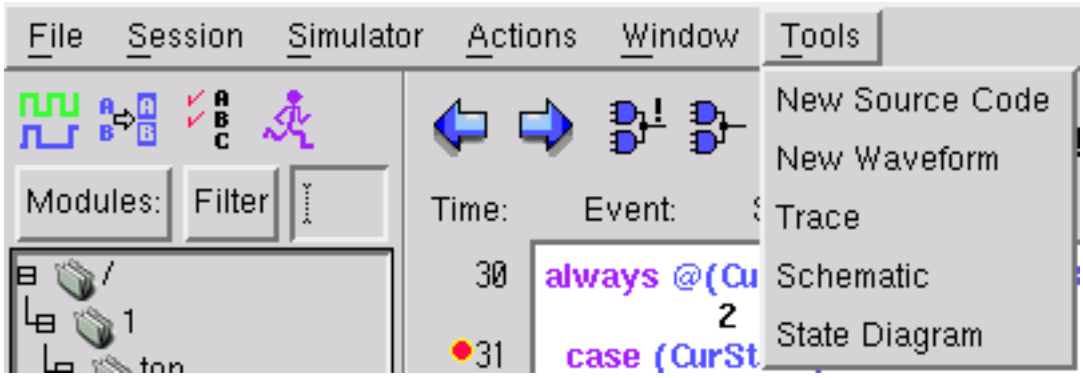


TABLE 60. Tools Menu Options

Menu Item	Description
New Source Code	Opens a new **source code** viewer.
New Waveform	Opens a new **waveform** viewer.
Trace	Opens the **Trace** Window.
Schematic	Opens the **Schematic** window.
State Diagram	Opens the **state diagram** window.
HDLLINT Report	Opens the **HDLLint reporting** tool.

Mouse Actions and Popup Windows

This section describes various popup windows and various mouse events, associated with the **Source Code** Window.

Breakpoints in Source Window

To set breakpoints, click the left mouse button in **Time** column of the **source code** window, in the left-most field at the desired line. Table 61, “Breakpoint Options”, is a summary of how to add and remove breakpoints.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 61. Breakpoint Options

Setting a Break Point	Click the left mouse button in the time column of the source code browser, if you do not want to set a breakpoint.
Disable a Break Point	Click the left mouse button on a red breakpoint, to disable the breakpoint. The breakpoint indicator changes from red to green.
Removing a Break Point	Press and hold the Control-key, and click the left mouse button, to remove the breakpoint.

Execution Marks Window

The blue and green markers on the far left indicate lines of code the have been executed in this interactive session, blue are the lines of code that have been executed, green are the lines of code that are currently being executed.

Execution Info In Source Window

Press the right button over any line of code where you want execution information: The information that is provided in the popup box is the following:

Line number
Number of times executed
Current execution status
Breakpoint status
VCT Branch status
VCT Multi Branch status

"How to" Operations for Source Code Window

This section describes how to interact with the source code window.

Selecting source code files

You can use either the command line, or the source code file selector to select the desired source code files. To use the source code file selector do the following:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File => Open =>Source code (import design)

This brings up the Select Files window, with a listing of directories and a list of files in the selected directory, select the files you want by double clicking with the left mouse button on the name of the file(s) or single clicking the left mouse button and selecting “Add File To List. (Note that you can select more then one file). If the file contains a list of files you need to manually insert a “-f” in front of the file name, if the file is a library you need to insert a “-v” in front of the file name. When you press the “Apply” button it will compile the source code that you have selected and load the top level design module in to your source code window

Showing and Locating source code for a module

To open the hierarchy, in the **source code browser** window (the little pane to the left of the **source code** window), left-click the **+** sign. The **+** is immediately to the left of the slash. The hierarchy expands to the level immediately below the current level.

To show the exact source code for each module, you can either:

- Drag and drop the name of the module into the source code window, or
- Toggle on the Modules button, and use the left mouse button to select the name of the module.

The **schematic** window highlights the selected module in reverse video. The source code associated with this module appears in the **source cod**e window.

Finding a signal assignments/use in source code

The list of signals for the selected module appear in the pane below the **module** pane.

1. Select a signal.
2. Click the **Variable** button.

   The source code window shows the first occurrence of that variable (signal).
3. Click the **Variable** button again.

   The source code window shows the first occurrence of that variable (signal), and so on.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Displaying a signal from source code window

Any time a module is selected, the variables of that module are displayed in the signal selector (the pane below the module pane). You can select a signal, or any number of signals, then with the middle mouse button held down, drag the signal(s) to the area of the waveform window which displays the variable names (left pane of wave window). Or you may press the “Display” icon (icon furthest to the left in the source code window). With the “Display” button toggled down, you can now select any number of signals and they will be automatically displayed in waveform window.

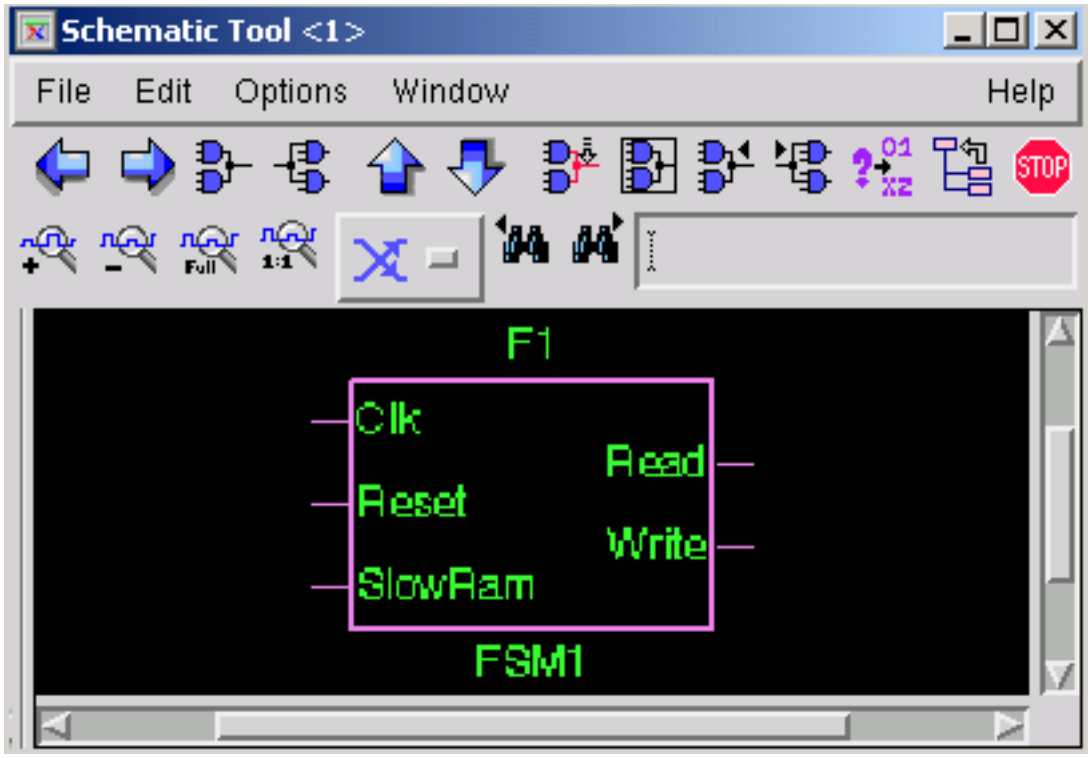
A third way to display signals in the waveform window from the source code window is by using a drag and drop directly from the source code window. From the source code window select with the middle mouse button any signal name, while holding the middle mouse button down drag the signal to the signal name area of the waveform window. When you release the middle mouse button while over the **Waveform** window, the selected signal will be displayed.

Finding source code from the signal window

In the Signal Window select the snap to source icon, it should darken Then use the middle mouse key to snap the cursor to the edge of the signal. The Undertow Suite Source Code window will display the exact line of code used to generate that particular signal edge.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Schematic Window



Description

The Undertow Suite automatically generates the schematic files from your source code files. These schematics can be gate-level or Register Transfer Level (RTL) schematics. You can use RTL schematics to see all sections of your RTL design graphically, and to quickly find drivers and loads from the **Schematic** window.

Use the **Schematic Browser** to browse through the graphical schematic. You can:

- Paste signals directly into the **Schematic Browser**, to display the gate driving it.
- Double-click on any input pin, to continue browsing through the gates in the current module.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Drag and drop signals from the **Module** or **Gate Browser** window, to any appropriate window.

  If a behavioral statement drives the output, and if you started Undertow through the **Interactive_tool**, then the **Schematic Browser** show a black box labeled *SOURCE*. If you click on the *SOURCE* box, a **Source** Window opens, and shows the line of source code that drives the output.

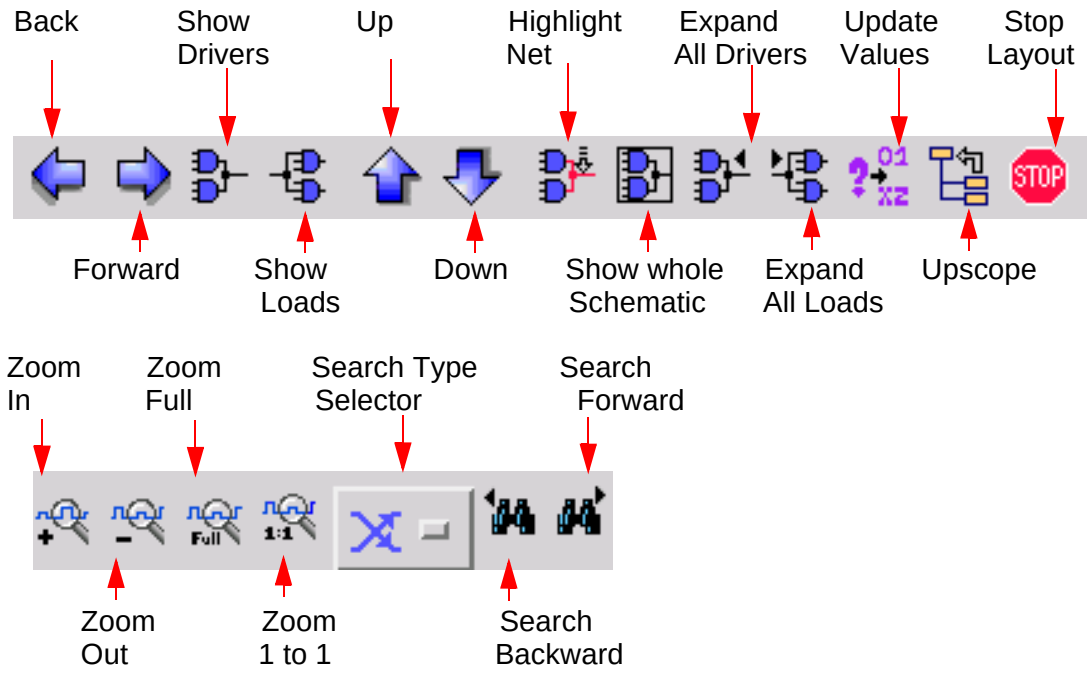
Starting the Schematic Browser Tool.

This section describes the Schematic window in the Undertow Suite. To view the schematic window from Undertow Suite, select either of the following pull-down menu items:

- ***Window* => *Tile Windows* => *Schematic***, or
- ***Window* => *Tile Windows* => *States and Schematics***

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Toolbar of Schematic Browser



Schematic Toolbar Icons

Table 62, “Schematic Toolbar Icons” describes the toolbar icons in the **Schematic browser**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 62. Schematic Toolbar Icons

Icon	Description
	Zoom Out/In—Zooms the schematic in or out.
	Zoom Full—Zooms the schematic to a full view.
	Zoom One to One—Displays all symbols at their normal size.
	Up & Down—The Up and Down arrows are enabled when you select a signal. Selecting Up or Down moves the zoomed-in schematic view, to show all signal connections in the schematic.
	Highlight Net—Displays the selected net in bright orange.
	Show Entire Schematic—Shows the entire gate-level or RTL schematic
	Expand All Drivers—Expands all drivers, one level above the current level.
	Expand All Loads—Expands all loads, one level above the current level.
	Stop Expansion—Stops generating large schematics.

 Copyright Veritools 2004 All Rights Reserved

TABLE 62. Schematic Toolbar Icons (Continued)

Icon	Description
	Search Signal—Sets the search item selector to one of the following: • Transition • Leading Edge • Trailing Edge • Value • String Use a text box next, in conjunction with the values or strings of the search signal item.
	Search Forward/Backward—Moves the cursor to the next or previous occurrence of the search signal item.

Schematic Menus

The Schematic Browser has five pull-down menus:

- File
- Edit
- Options
- Window
- Help

The following subsections describe these menus and their submenus.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Menu

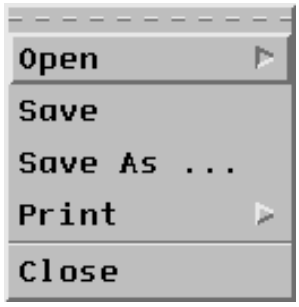


TABLE 63. File Pull-down Menu

Menu Item	Description
Open	Pops up a submenu to select the type of schematic file to open.
Save	Saves the schematic, and uses the existing file name.
Save As...	Saves a new schematic, or saves the existing schematic with a new file name.
Print	Pops up a submenu of options for printing a schematic.
Close	Closes the schematic window

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Open Submenu

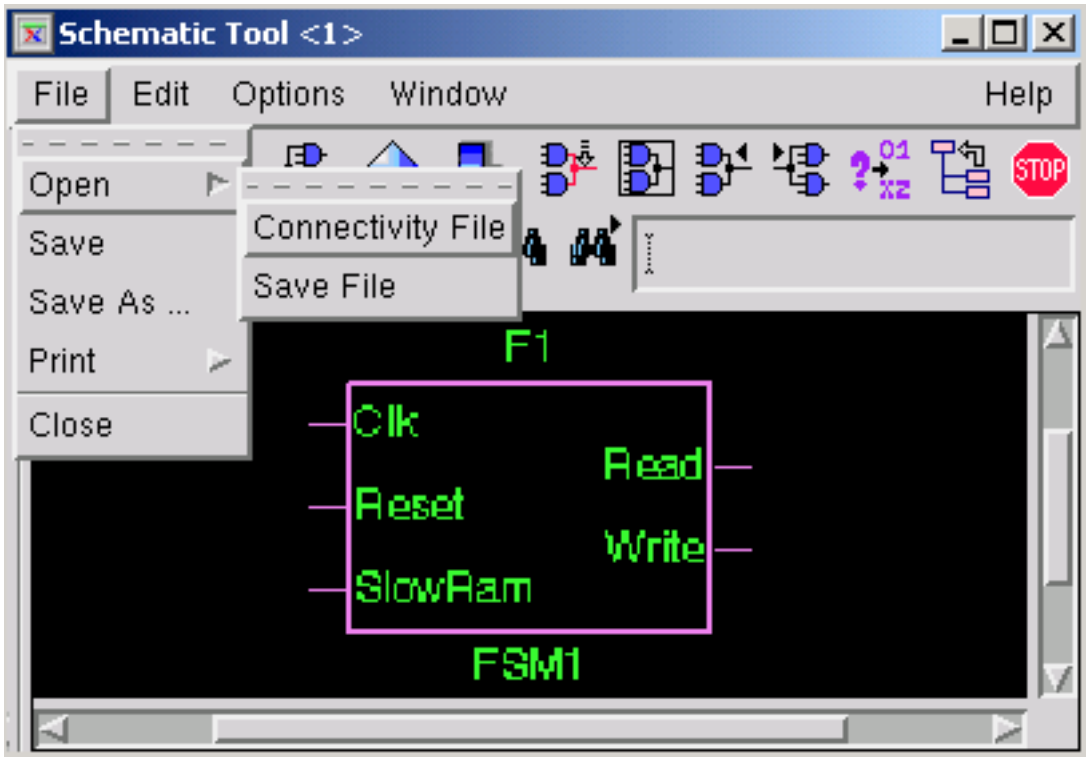


TABLE 64. File Open Submenu

Menu Item	Description
Connectivity File	Opens a new schematic.
Save File	Saves the current schematic.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Print Submenu

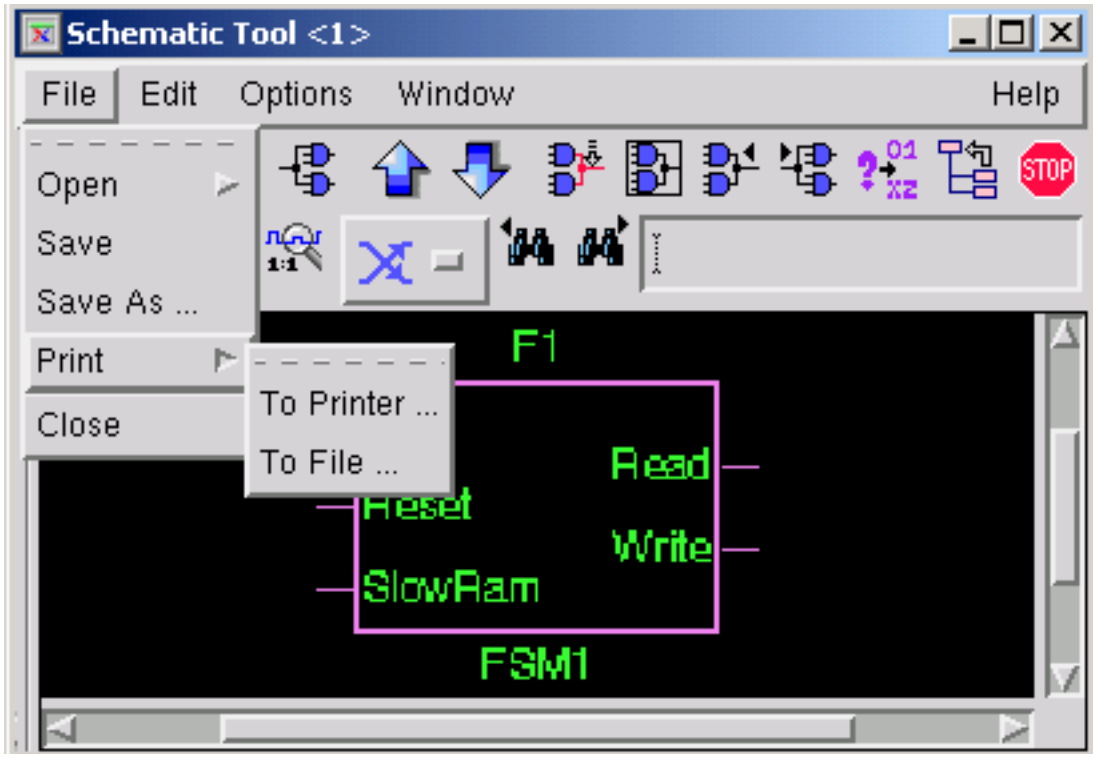


TABLE 65. File Print Submenu

Menu Item	Description
To Printer...	Opens a dialog box, in which you select the name of the printer, to print the schematic that is currently displayed in the **Schematic Browser**. *Note: The printer output is in Encapsulated Postscript (EPS) format.*
To File...	Opens a file dialog box, in which you select the name of the file that stores the print image. *Note: The print file is in Encapsulated Postscript (EPS) format.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Edit Menu

TABLE 66. Edit Menu Options

Menu Item	Description
Select All	Select the whole schematic
Cut	N/A
Copy	N/A
Paste	N/A
Clear	Erases the contents of the **Schematic** window

Options Menu

TABLE 67. Options Menu

Menu Item	Description
Create New Window	Open a separate new window
Create One Window	Create One Window
Display	Automatically display waveform of the signal if selected.
Annotation	Names - Will display the signal/module name if selected.
	Value - Auto Update - auto updating the value
	Binary - Display the binary value.
	Hexadecimal - display hexadecimal value
List Hierarchical Names	List all the hierarchical name for the selected signal
Bundle Connections	Bundle all the bus connections
Single Scope Only	limits the display current level of hierarchichy
Enable Results Detail Window	Open results detail Wiondow
Show Maximum Details	Show flattened schematic
Logic Back-Tracking	N/A

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 68. Icons Menu

Menu Item	Description
Zoom In	Enlarges the scale by a factor of 2X (twice its current size).
Zoom Out	Shrinks the scale by a factor of 2X (half its current size).
Zoom Full	Zooms all the way out.
Back	Previous view (to be implemented).
Forward	Next view (to be implemented).
Display Current Drivers	(To be implemented).
Show Drivers	Shows the drivers for the selected signal.
Show Loads	Show the loads for the selected signal.
Up	Shows the next driver or load.
Down	Shows the previous driver or load.
Highlight Net	If you toggle this option *on*, any signal that you drag and drop into this window is automatically highlighted.
Show Whole Schematic	Shows the entire schematic.
Expand All Drivers	Expands all drivers.
Expand All Loads	Expands all loads.
Stop Layout	Stops the Schematic drawing. You might need to use this option for very large designs that load slowly and require a lot of memory.
Upscope	Moves up one scope in the schematic window.
Display	If you toggle this option *on*, when you select a signal in the **Schematic** window, the waveform for that signal displays in the **Waveform** window.
Update	If you toggle this option *on*, the **Schematic** window immediately updates values on signals. For very large designs, you might want to disable this option, so the tool runs faster. After simulation finds a specific time point, you can enable this feature again.
Names	Shows the names of a selected net, including its complete path through the hierarchy.
Bundle Connections	Bundles all wires together.
Single Scope Only	Shows the schematic for *only* the current module.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Window Menu

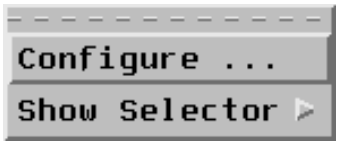


TABLE 69. Window Menu

Menu Item	Description
Configure...	Opens a dialog box for configuring the **schematic browser** .
Show Selector	Shows the saved **Schematic** windows.

Configure Dialog Box

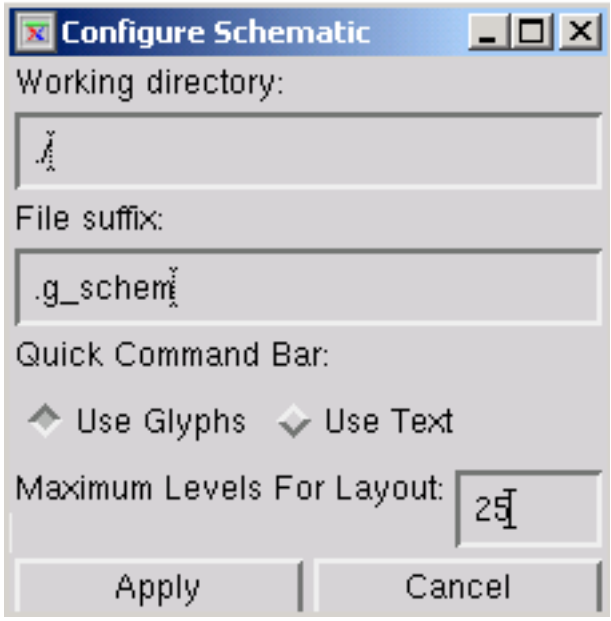
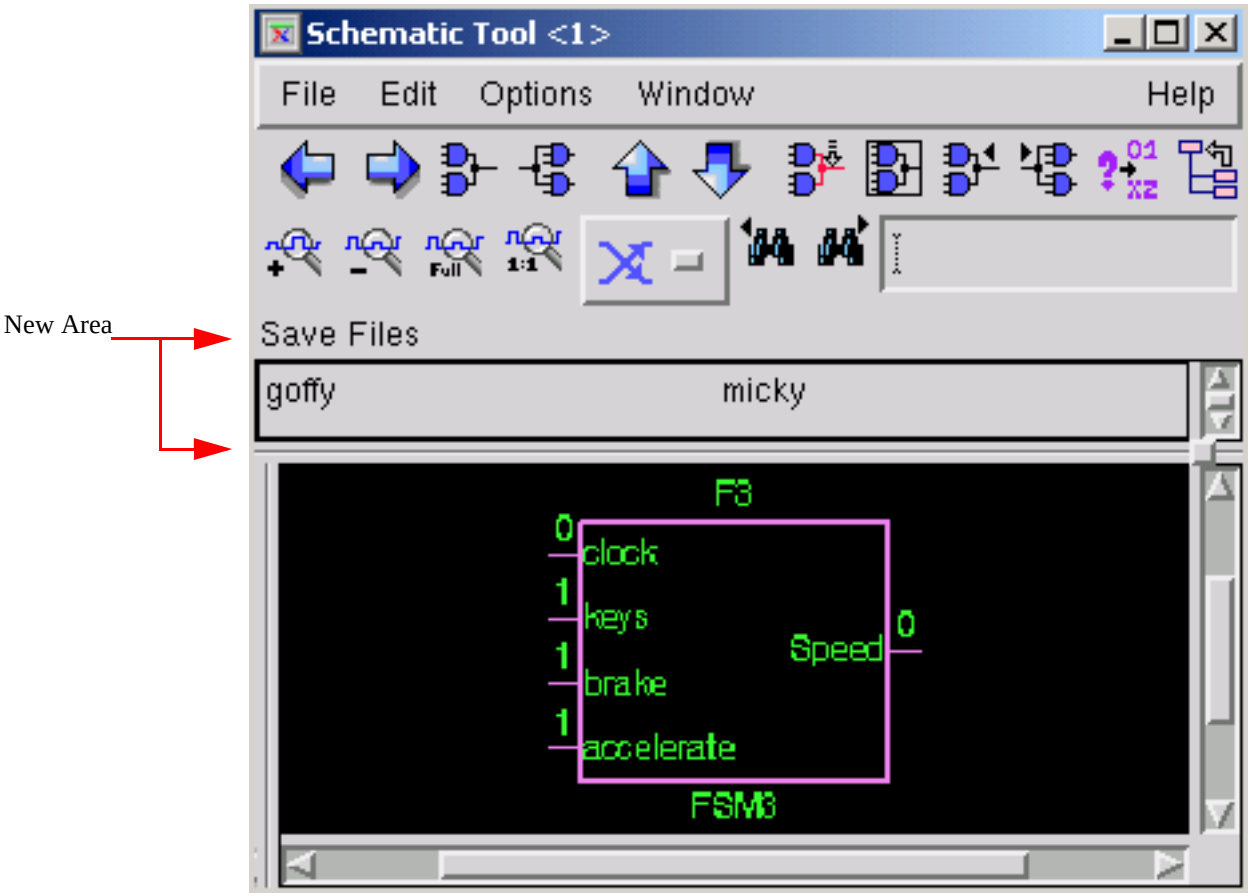


TABLE 70. Gate Browser Configuration Dialog

Item	Description
Working Directory:	Specifies the directory where the file exists.
File Suffix:	Appends the specified suffix to the file name.
Use Glyphs	Turns on pictures (icons) to display on buttons.
Use Text	Uses *only* text to label buttons.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Show Selector



The **New Area**, annotated in the figure above, lists all of the saves captured from the ***File*** => ***Save As*** menu item.

0.1 HDL Analyzer

Undertow draws a complete schematic of your Verilog HDL source code. Using the HDL Analyzer, a schematic is drawn with RTL primitives.that are;

2 input MUXs representing "ifs", and "else_ifs"
N Input MUXs representing "case" statements
AND, NANDS for scalar or vectors
ORs, and NORS for scalar or vectors
ALIAS gates for assignments
CONCATENATE gates for concatenate operators for combining signals
FFs, and Latches just as with a synthesizer
COMP_EQUAL for the "==" operator
APLUS, AMINUS, and others,

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

see list below for gates that support all of the Verilog operators;

2CONCAT.gate
ADIV.gate
AMINUS.gate
AMOD.gate
AMULT.gate
APLUS.gate
COMP_EQU.gate
COMP_GT.gate
COMP_GTE.gate
COMP_LT.gate
COMP_LTE.gate
COMP_NEQU.gate
CONCAT.gate
COND.gate
FUNC.gate
LAND.gate
LOR.gate
RAND.gate
ROR.gate
RXNOR.gate
RXOR.gate
SLEFT.gate
SRIGHT.gate
UBEQU.gate
UMINUS.gate
UNOT.gate
UOR.gate
UPLUS.gate
URAND.gate
URNAND.gate
URNOR.gate
URXOR.gate

- The schematic is drawn either as a block diagram form (using "initial", always", "FF", or "continuous assignment" blocks), or as a detailed view with the above gates to describe the RTL source code as a schematic drawing.
- For example the user can see any module on the schematic and then double click into this module to see the schematic details inside of this module.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- If this module has "initial", always", "FF", or "continuous assignment" blocks, these will be displayed as these type of blocks unless "Show Maximum Detail", is turned on.

- If "Show Maximum Detail" is turned on or the user double clicks into any of the "initial", always", "FF", or "continuous assignment" blocks, the RTL equivalent gates described above will be used to describe the RTL circuit on the schematic view.

- Users can drag and drop any signal, any module from the source code window, waveform window or signal selector to the schematic window to see the views of this element from a schematic stand point. Users can drag and drop any element from the schematic window to the source window in order to see this element from the schematic as an enter in the source code on the source code window.

1. Here is a short description of the users interface

   a. Show schematic details

   b. Drag and drop a wire, signal or module onto the schematic window

   c. Double click on the empty area of any module, always block, continuous assignment or initial block to see the synthesized RTL in that block.

   d. If any pin inside of block is high lighted, then the schematic only traces the logic from that pin through block to opposite pins in block.

   e. If user double clicks on any pin inside of always block, continuous assignment or initial block, then just the drivers or loads on that pin are shown, logic back tracking stops at one level. Users can then use "Show all Drivers", or "Show all Loads" in order to see logic drawn on each subsequent level.

   f. Users can use Options => Create New Window to create as many schematic windows as they want and can drag and drop any signal, block or gate from one window to the next.

   g. Users can press the Back or Forward icons to see the last schematic screen back or forward.

   h. Users can press "Upscope" to see the scope one level above the scope where they are located. If they have high lighted any signal, this signal stays high lighted in the "Upscope" view also.

   i. User can drag any element from the schematic to the source code window in order to see the source code for that element, elements include wires or signals, blocks, or any gate including any RTL primitive gate.

   j. Users can turn High light new on and drag any signal to schematic to see that signal high light if it is this schematic view. Users can also use the left mouse button when High Light net is on to high light any net.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

k. When any net is high lighted, pressing Up and Down arrows will bring into the center of the screen one at a time the connections on this net.

Trace Back

1. Users can trace any net backward or forward using the Show All Drivers or Show All Loads icons, or double clicking on the net, note the net glows when you are in a position to double click on it.

NOTE: When users have a box block to view, i.e., always, continuous assignment or initial block, clicking on the name inside dives down one level and show connection for one level of elements connected to this pin, double clicking on outside pin on box, connects net or disconnects net to this pin.

"How to" Operations, for the Schematic Browser

This section describes how to operate the **schematic browser**, and how it interacts with the other windows in the Undertow Suite.

Tracing a Signal in the Schematic Window

1. To open the **schematic** window, do either of the following:
   - In the Undertow **waveform viewer**, select the **Schematic** icon, or
   - In the Undertow Suite **Source Code** window, select ***Window*** => ***Tile Window*** => ***Schematic***.
2. Select a signal in one of the following windows:
   - Source Code
   - Choose
   - Undertow (waveform)
3. Use the middle mouse button to drag and drop the selected signal into the **schematic** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

All signal drivers are displayed.

4. To trace a signal driver in the schematic, double-click the left mouse button on the signal name.

Highlighting a Net in the Schematic Browser

1. To enable the *highlight net* function, click the **Highlight Net** icon.
2. Drag and drop any signal onto the schematic, from any of the following windows:
   - Source Code
   - Signal Selector
   - Waveform

   The schematic highlights the selected net in orange.
3. Click the *up* or *down* arrow icon buttons.

   These buttons move up or down the schematic hierarchy, to show each connection of this net to every element in this highlighted network.

Locating a Schematic Element in a Source Code Element

Drag any element from the schematic, into the **source code** window, to display the source code for this element.

Displaying a Signal from the Schematic Window

Drag and drop any signal from the schematic, into the **waveform** window, to display the waveform for this signal.

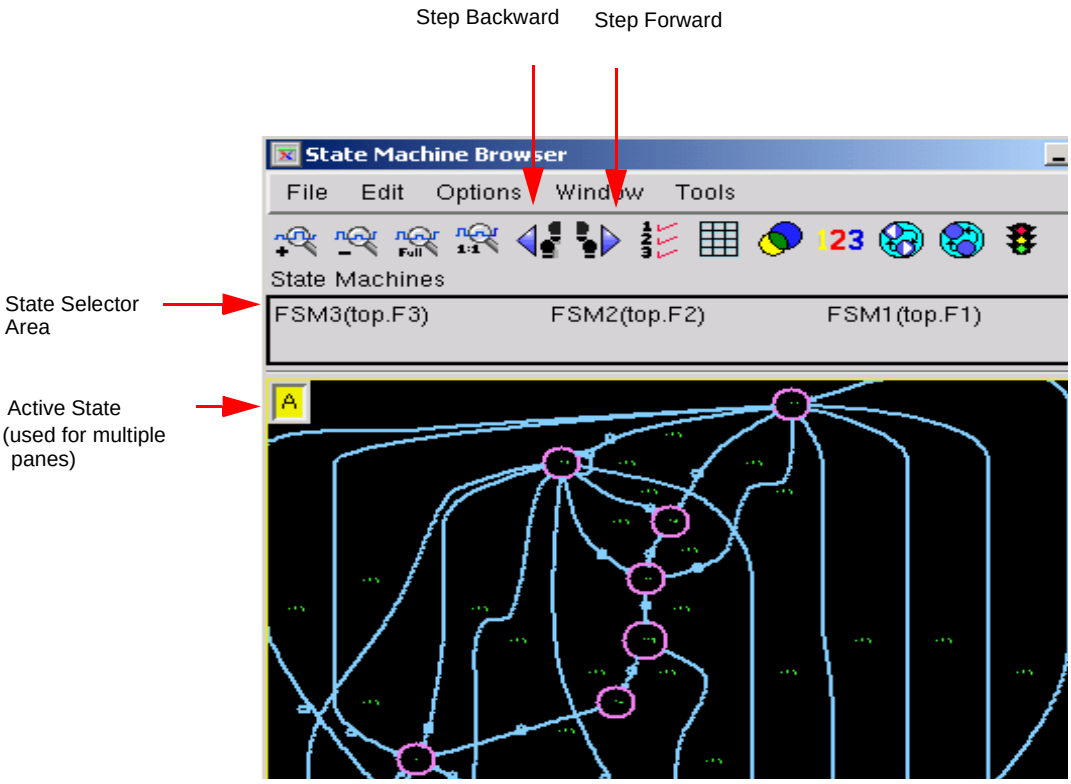
Finding the Source Code from the Schematic Window

1. In the **Schematic** Window, place the cursor in the center of the schematic graphical symbol.
2. Click the left mouse button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The Undertow Suite **Source Code** window displays the exact line of code that created the selected schematic block.

State Machine Browser



Description

The Undertow Suite includes a complete state coverage tool, so you can see exactly which states have been tested so far.

1. Drag and drop State Machine blocks from the RTL **Schematic** Window, into the **State Diagram** Window.

The **State Diagram** Window shows the exact state diagram.

2. Use the **Source Code** Window to view the lines of source code that correspond to this state diagram.

You can use the **State Diagram** feature to:

- View and step any state machine.
- View and step multiple state machine at the same time, to see the interaction between these state machines.
- View state machine interaction within the same time point.
- Select a single state on any diagram, and fully explore what other state machine are doing during this state.

Starting the State Tool

To open the state browser from the source code window, select the ***Window*** => ***Tiles*** => ***Tools*** menu item from the pull-down menus.

State Toolbar

Table 71, “State Toolbar Icons” describes the toolbar icons in the **State** browser.

TABLE 71. State Toolbar Icons

Icon	Description
	Zoom Out/In—Zooms the schematic in or out.
	Zoom Full—Zooms the schematic to a full view.
	Zoom One to One—Displays all symbols at their normal size.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 71. State Toolbar Icons (Continued)

Icon	Description
	Step—Steps forward to the next state, or back to the previous state.
	Persistence—Marks all states where the **State Machine Browser** has stepped.
	State Table—Shows what states might be transitions from other states.
	State Coverage—Shows both the states that have been covered or traversed, and the paths have been traversed.
	State Sequence—Shows you exactly how the sequence of State Diagrams occurs.
	State Coincidence, Step—Lets you select one or more states, on any number of state machines. When you click the **Step Forward** or **Step Backward** icons, the state diagrams step *only* to those states where the selected states are also enabled or *True*.
	State Coincidence, Display—Pops up a window that shows state coincidences.
	Go-to-State—Selects a state. If you click on this state, it appears in the **waveform** viewer.

State Menus

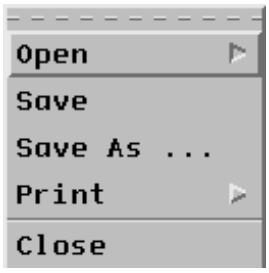
The State Viewer includes the following size pull-down menus:

- File
- Edit
- Options,

- Window
- Tools
- Help

The sub-sections that follow describe these menus, and their submenus.

File Menu



File Pull-down Menu

TABLE 72. File Pull-down Menu

Menu Item	Description
Open	Pops up a submenu (see Section , “File Open Submenu”).
Save	Save the specified State Machine to an existing file.
Save As...	Save as a new file.
Print	Pops up a submenu (see Section , “File Print Submenu”).
Close	Closes the **State Viewing** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

File Print Submenu

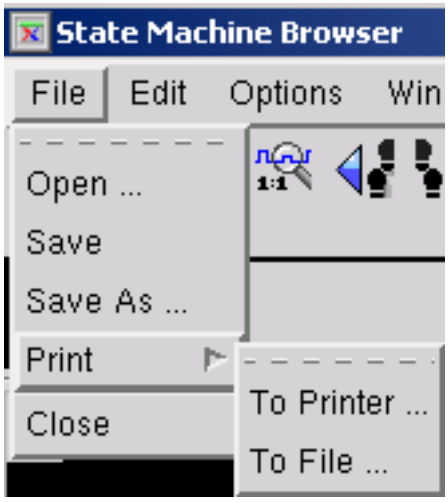


TABLE 73. File Print Submenu

Menu Item	Description
To Printer...	Opens a dialog box, in which you select the name of the printer, to print the state machine that is currently displayed in the **State Browser**. *Note: The printer output is in Encapsulated Postscript (EPS) format.*
To File...	Opens a file dialog box, in which you select the name of the file that stores the print image. *Note: The print file is in Encapsulated Postscript (EPS) format.*

Edit Menu

TABLE 74. Edit Menu Options

Menu Item	Description
Select All	Select all states in the FSM Browser
Cut	N/A
Copy	N/A
Paste	N/A
Clear	Clears the selected pane.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Options Menu

TABLE 75. Options Menu

Menu Item	Description
Overdraw Mode	Double up the Pixel so it can be view best with projector(demostration purpose)
Load Step File	Load the step file

TABLE 76. Options Menu

Menu Item	Description
Zoom In	Enlarges the scale by a factor of 2X (twice its current size).
Zoom Out	Shrinks the scale by a factor of 2X (half its current size).
Zoom Full	Zooms all the way out.
Zoom 1:1	Displays all symbols at their normal size.
Step Back	Steps backward through the state machine.
Step Forward	Steps forward through the state machine.
Persistence	Marks all states where the **State Machine Browser** has stepped.
Over Draw Mode	Makes the diagram bolder.
Load Step File...	Loads a new `.def` file.
State Table	Shows what states might be transitions from other states.
State Coverage	Shows both the states that have been covered or traversed, and the paths have been traversed.
State Sequence	Shows you exactly how the sequence of State Diagrams occurs.
State Coincidence, Step	Steps through coincidences.
State Coincidence, Display	Displays coincidences.
Go to State	Moves to the selected state.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Window Menu

TABLE 77. Window Menu

Menu Item	Description
Configure...	Opens a dialog box for configuring the **state machine** browser
Show Selector	Toggles the **SM Selection** area on and off.
Add Pane, Vertical	Adds a vertical pane, for viewing multiple state machines.
Add Pane, Horizontal	Adds a horizontal pane, for viewing Multiple state machines.
Remove Pane	Removes the selected pane.

"How to" Operations, for State Machine Browser

This section describes how to operate the State machine browser window, and how it interacts with other windows in the Undertow Suite.

Loading a Module

- The top pane of this window is the State Diagram Selector pane.
- The bottom pane of this window is the state selector area.

Undertow Suite automatically extracts the state machine diagrams from your design, and populates the state selector area with a complete list to choose from.

1. To make a selection, click the left mouse button on the desired module.

   The desired diagram appears in the viewing area.

Multiple Views

To display more than one state diagram, do the following:

1. Select one of the following from the pull-down menus:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- ***Window => Add Pane, Vertical***, or
- ***Window => Add Pane, Horizontal***

Choose either the vertical or horizontal pane, so the state diagram fits properly.

The new pane is created.

2. Set the new pane as the active view (see Section , “Setting the Active View”).
3. To select a module, click on the module name in the state selector area.

   The new pane displays the state machine diagram for the selected module.

Setting the Active View

If the State Browser displays more then one view, as shown in Figure 149, “Multiple State Machine Views”, you can select the active pane

1. On the pane to activate, toggle the *active* indicator, denoted by the **A** button in the upper-left corner of its pane.

   The button turns yellow on the active pane.

Note: Only one pane can be active at any time.

Stepping the State Diagram

To step the active pane that holds a state, click on the **Step Forward** or **Step Backward** arrow icons, on the **state diagram** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

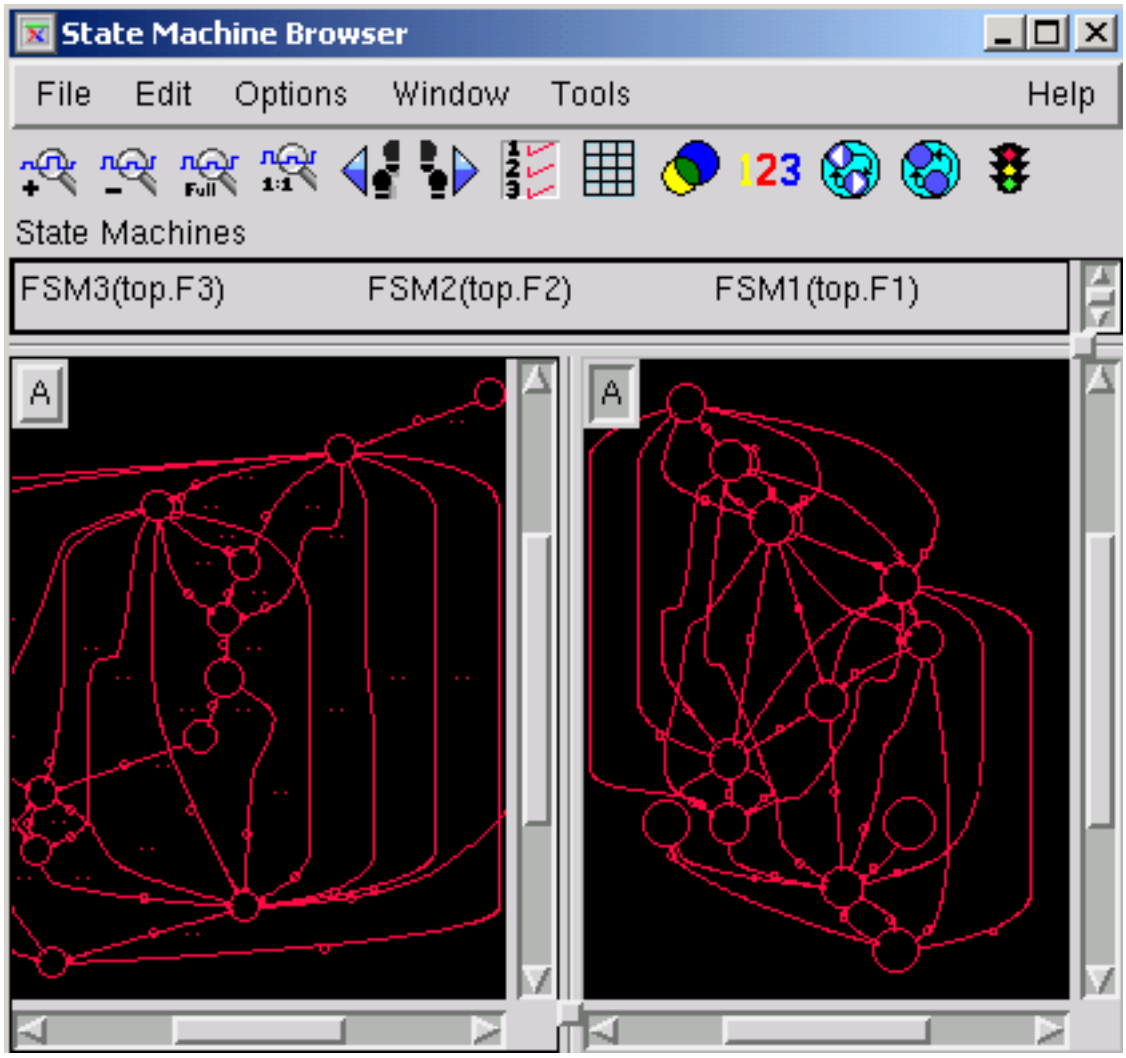


FIGURE 149. Multiple State Machine Views

Stepping the State Browser

To step the active pane, use the step forward (back) icons in the toolbar.

- The **State Browser** shows the next (previous) state.
- The **State Browser** also updates the corresponding state for all other state machines that you are viewing.
- The **source code viewer** and the **waveform viewer** also step, to remain synchronized with the **state browser**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The reverse is also true: if you step the **source code viewer**, the **State Viewer** steps accordingly.

Finding Source Code from the State Window

1. Use the middle mouse button to drag and drop a state bubble from the **State Window**, into the **source code** window. \

   The Undertow Suite **Source Code** window displays the exact line of code that generates the selected state bubble.

Showing Persistence

1. Toggle on the **Persistence** icon (see Table 71, “State Toolbar Icons”).
2. Step the state diagrams, from any time in the simulation, to any other time.

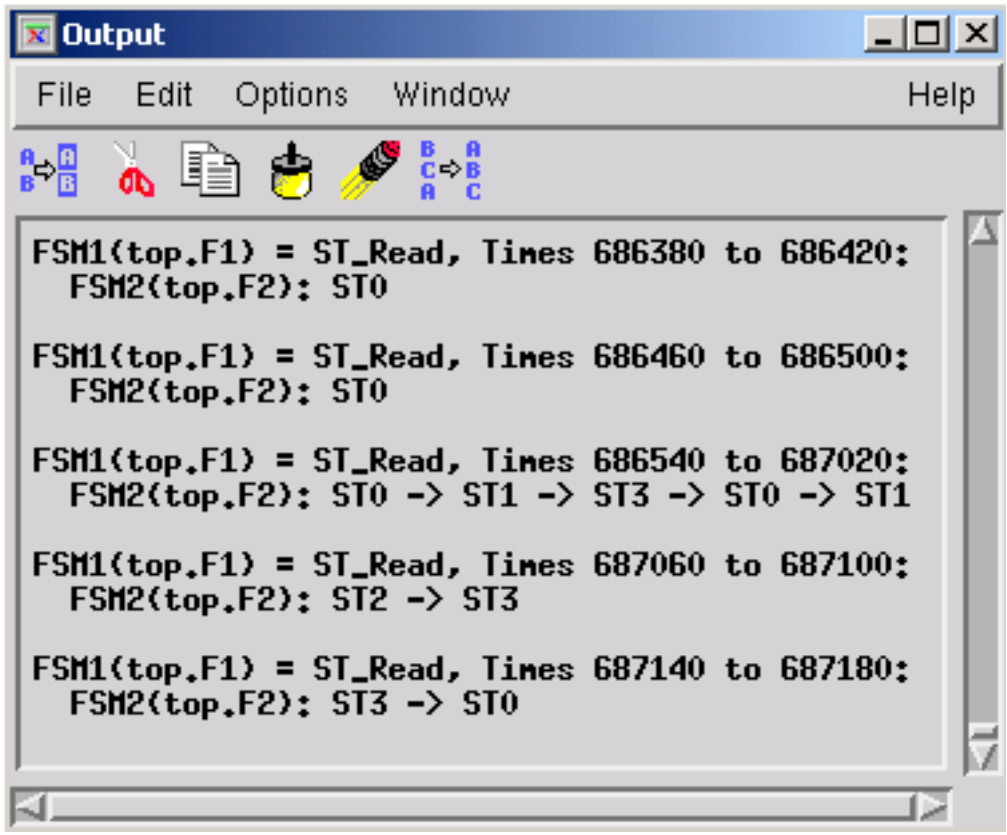
You can see exactly which states the simulator has exercised during this time.

Analyzing Coincidence

1. Toggle on the state **Coincidence** icon (see Table 71, “State Toolbar Icons”).
2. Select any state in the pane that is active.

The output display window lists the states in all other state diagrams, that are coincident to this state.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Trace Window

The **Trace** window provides powerful behavioral debugging.

- The top of the window has a label, showing the simulation time for the last **Trace** event stored.
- Below this is a simulation time line.

  1. Move the thumb dial of this time line.

     The lower section of the window displays the contents of the source file, for the first event in the newly-selected time.

     The line that is currently executing, is highlighted. B

- Below this time line is an event line.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Move the thumb dial of this event line.

   The lower section displays the source file for the newly-selected event.

Note: The Trace window shows events in the order that the simulator executed them. This sequence helps you identify and solve errors due to race conditions. A label, under the event scale, shows how many events occurred during the currently-selected simulation time.

Beneath the event scale is a row of debugging buttons. Each button has a picture, which indicates its function. As the mouse cursor enters a button, the function of the button appears in the area at the right of the button row.

1. For a more detailed description of the button's function, press and hold the right mouse button on the button's picture.

   A pop-up window describes the function of the button.

Below is a description of the actions for each button, in the order they appear (from left to right) in the button row (icon bar).

Step Backward—When you click this button, the **Trace** Window steps backwards one step through the history. Even if you hold down the button, it steps back *only* one event.

Step Forward—If the last event is currently displayed, when you click this button, the simulator advances forward one simulation event. Otherwise, the **Trace** Window advances forward one event through the history. Even if you hold down the button, it steps forward *only* one event.

Turn Animation On—This button toggles the *Animation* mode of the **Trace** Window. If Animation is enabled, the **Trace** Window updates the **source** window with the current file, and the line that is executing as the simulation progresses. If you disable Animation, the **Trace** Window updates *only* its simulation time label. The **Trace** Window always maintains a history of all simulation trace events. This button displays with either a green or red border. The color of the border indicates the action that occurs when you click on the button. For example, if the border is green, then clicking the button ENABLES Animation mode. If the border is red, then clicking the button DISABLES Animation mode. The area to the right of the button row shows you which functions are activated when you click the button.

Set Scope—When you click this button, the simulator sets the scope to the event that currently appears in the source pane of the **Trace** Window. The **Trace** Window displays the scope of the current event after the *Instance:* label.

 Copyright Veritools 2004 All Rights Reserved

Inspected Selected Signal—Click this button after you select a signal from the source pane in the Trace Window. To select a signal, double-click on any signal in the source pane. After you select a signal, click on the selected signal in the Inspector (if it is not already visible), and place the selected signal in the Inspector signal text field. You can use the Inspector to obtain either the signal value from the monitor history, or the value at the current simulation time.

Modifies Selected Signal—Click this button after you select a signal from the source pane in the Trace Window. See the ***Inspected Selected Signal*** description above for details about how to select signals. Clicking this button pops up the **Alter Signal** Window. Place the selected signal in the signal text field of the Alter Signal Window. You can use the **Alter Signal** Window to change the current value of a signal.

Display Current Drivers—Displays the current drivers of your selected signal.

Breakpoint On Line—Sets a line breakpoint on your selected line, in the **Trace** source pane.

Breakpoint On Signal—Click this button after you select a signal from the source pane in the Trace Window. See the ***Inspected Selected Signal*** description above for details about how to select signals. Clicking this button pops up the **Breakpoint Editor** Window. Place the selected signal in the expression text field. In the **Breakpoint Editor**, you can use complex expressions to define breakpoints.

GoTo—When you click this button, a dialog box pops up. You can enter a time value in this dialog. The Trace Window uses the time that you enter, as the simulation time.

Simulate For A Time—When you click this button, a dialog box pops up. You can enter a time value in this dialog. The time that you enter is the amount of time to simulate, and then stop.

Open A Source File—Loads your selected module into the **source** window.

Show Status—By default, this icon is toggled on. When this icon is on, the **Trace** window shows information, such as the name of the trace file, the current trace line number, and the outcome of the current trace line. If you toggle this icon off, then the Trace window does not show this information.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Application Notes

Integration with Gate level netlist

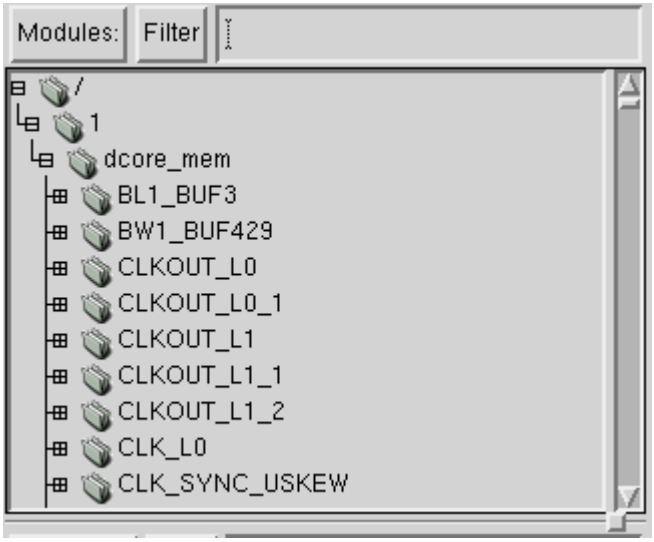
A new feature in Undertow allows designers to load in gate level design and debug the problems.

Couple of features has been implemented and more will be available in the future release.

To Load in the gate level netlist, simply type in ut -iv <design>.v <library>.v

Undertow window should open, expand the hierarchy using the hierarchical brower on the left.

FIGURE 150. Hierarchical Browser



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To see the schematic, select the module and click on the right mouse button.

Note: Please make sure to select the module that you want to read. Simply loading in the top module and nevigate to the desired module may take some times as gate level contains millions of gates, it will take a while to show the whole schematic.

To show logic across the boundry:

There are 2 ways to show the logic across boundry(hierarchy):

1. Selected the boundry pin, drag and drop the pin into a blank area

2. The schematic will show where this pin is connected to the other module

3. Select the pin on that module and click it.

4. Now the path across the module is shown

Or

1. Select the boundry pin,

2. Options->show maximum details

3. the path will be show despite the border as if boundry does not exist.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 8

Perl Scripting

Script Tool

To Download Undertow Perl Package

Perl Package is available at the release download page.

http://www.veritools.com/download_latest.shtml

For Solaris the archive is named perl_Tk_solaris.tar.gz

Download it and install it with the instruction below:

To Install Perl package

We have not bundled Perl or Perl/Tk (Tk is the Perl module which provides the GUI services) with Undertow because of their size. Instead we have archived the directory /usr/local/lib/perl5 in a file and placed this file on our web site. /usr/local/lib/perl5 is the default installation directory for Perl5. Perl/Tk resides inside /usr/local/lib/perl5.

For Solaris the archive is named perl_Tk_solaris.tar.gz

NOTE; If you download it with Netscape then Netscape will automatically unzip it and leave you with the file perl_Tk_solaris.tar.

You can gunzip the archive with the command:

gunzip perl_Tk_solaris.tar.gz

You must then un-tar the archive with the command:

tar xvf perl_Tk_solaris.tar

This creates the directory perl5/ in the current directory.

If you do this as root while you are sitting in /usr/local/lib, you are done. If you do this somewhere else, you then need to do one of the following in order to use the perl5 and perl/Tk you have just untarred:

1. You can move the newly created perl5/ directory into /usr/local/lib

2. You can make a link to the newly created perl5/ directory from /usr/local/lib

3. You can add the location of the new perl5/ directory to the environment variable PERLLIB. This will cause perl to search the specified location before searching /usr/local/lib/perl5 for modules such as Tk.

Therefore, supposing you have issued the "tar xvf ..." command from a directory called "/usr99/abcde".

1. To move the newly untarred perl5/ directory into /usr/local/lib, do the following (you will need to be root):

cd /usr/local/lib

cp -rp /usr99/abcde/perl5 .

2. To make a link instead of copying, enter the following (again, as root):

cd /usr/local/lib

ln -s /usr99/abcde/perl5 .

3. To leave /usr/local/lib/perl5/ alone and point undertow to the newly untarred perl5/ directory, set the environment variable PERLLIB by entering the following long line before running undertow (as yourself):

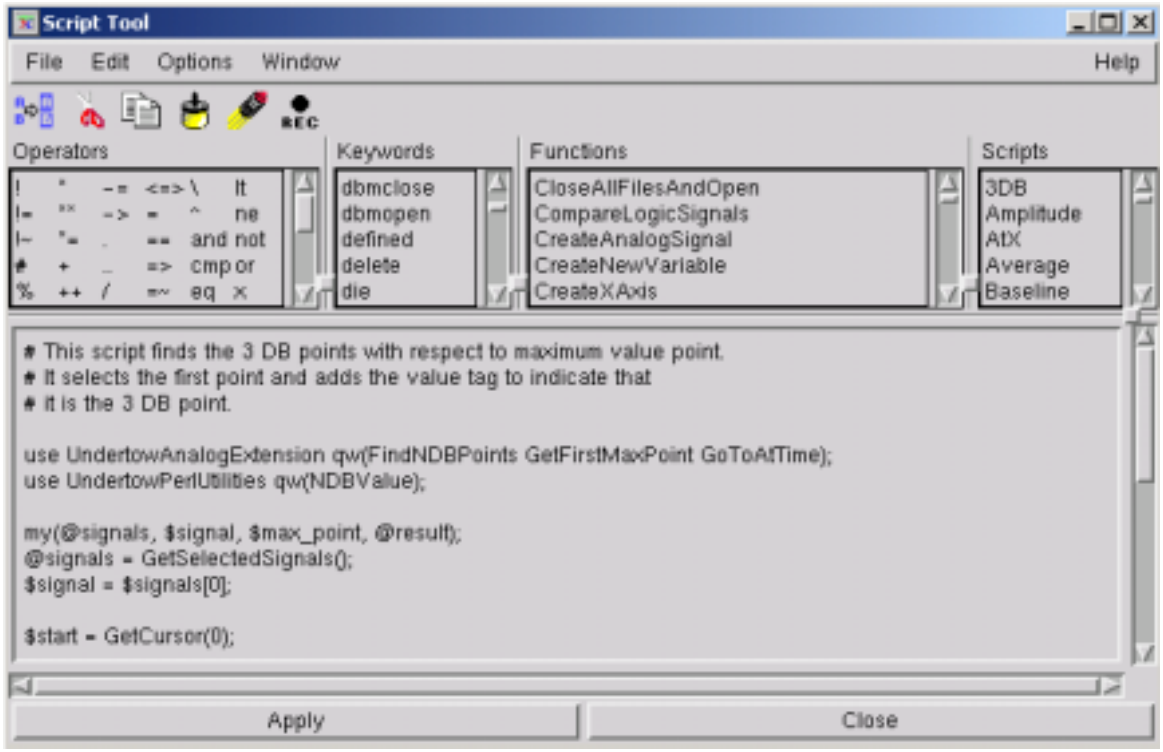
setenv PERLLIB /usr99/abcde/perl5/5.6.1/sun4-solaris:/usr99/abcde//perl5/5.6.1:/usr99/abcde/perl5/site_perl/5.6.1/sun4-solaris You are now ready to use perl/Tk and the builder script.

Script Tool

With Perl installed, You can use the Undertow Script Tool to create and run your own Perl scripts.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 151. Script Tool Window



In the list boxes, Undertow provides all of the operators and keywords for the Perl language. Those boxes also provide Undertow-specific functions, and a list of the scripts in your **run** directory.

Perl Configuration

In order for the scripting tool to work correctly Undertow needs to access the scripts directory and the Perl program.

Installing Perl

Perl is constantly being updated and is accessible to the public, so it is important to use the correct version of Perl expected by these scripts. Make sure Perl 5.00502 is installed in a location accessible to Undertow. And the Perl environment variable must be configured for this location as follows:

setenv PERL5LIB <perl_installation_dir>5.00502

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Accessing the Scripts Directory

Bring up the scripting window by using selecting ***Tools=>Script Tool*** in the WaveForm Window. In the Script Tools Window use ***Window=>Configure=>***Working Directory field to set the location of the scripts directory. usually *ut8.3.3/scripts*. By default the Working Directory field is set to the UT_WORK_DIR, which may be configured differently.

Overview of the Perl Language

Perl is an interpreted language, optimized for:

- Scanning arbitrary text files.
- Extracting information from those text files.
- Printing reports based on that information.

Perl is also a good language for many *system management* tasks. It combines some of the best features of:

- C
- send
- ask
- sh

If you are familiar with those languages, you should have little or no difficulty using Perl.

Commands

- # comment text

  Perl considers any command that begins with # as a comment, and does not process it.

- Printing uses the following format:

  print "\nThe time from low to high = *$delta*\n";

  Perl prints the **value** of *$delta* in its place.

- References to scalar variables always begin with **$**, even when referring to a scalar that is part of an array. For example:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
$days          # a simple scalar variable
$days[28]      # 29th element of array @days
$days{'Feb'}   # one value from an associative array
$#days         # last index of array @days
```

- Entire arrays or array slices are denoted by **@**:

```
@days              # ($days[0], $days[1],... $days[n])
@days[3,4,5]       # same as @days[3..5]
@days{'a','c'}     # same as ($days{'a'},$days{'c'})
```

- Entire associative arrays are denoted by **%**:

```
%days              # (key1, val1, key2, val2 ...)
```

- In Perl, you can treat a sequence of commands as one command, or *block*. To do this, enclose the entire block in braces:

  {}

  For example, you can use the following compound commands to control flow:

```
if (EXPR) BLOCK
if (EXPR) BLOCK else BLOCK
if (EXPR) BLOCK elsif (EXPR) BLOCK ... else BLOCK
LABEL while (EXPR) BLOCK
LABEL while (EXPR) BLOCK continue BLOCK
LABEL for (EXPR; EXPR; EXPR) BLOCK
LABEL foreach VAR (ARRAY) BLOCK
LABEL BLOCK continue BLOCK
```

  Note: Unlike C and Pascal, these are defined in terms of BLOCKs, not statements. This means that the braces are required—Perl does not allow dangling statements.

- The **while** statement executes the block, as long as the expression is true (does not evaluate to the **null string** or **0**).

- The **LABEL** statement is optional, and if present, consists of an identifier followed by a colon. The **LABEL** identifies the loop, for use in the loop control statements:

  - next
  - last
  - redo

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- A continue BLOCK, if present, always executes *just before* Perl is about to evaluate the conditional again. This is similar to the third part of a **for** loop in C. So you cannot use this statement to increment a loop variable, even if you used the **next** statement (similar to the C **continue** statement) to continue the loop.
- If you replace the word **while** with the word **until**, the sense of the test is reversed, but Perl still tests the conditional before the first iteration.
- In either the **if** or the **while** statement, you can replace **(EXPR)** with a BLOCK. The conditional is true if the value of the last command in that block is true.
- The **for** loop works exactly like the corresponding **while** loop:

```
for ($i = 1; $i < 10; $i++)
{
 ...
  }
```

is the same as

```
$i = 1;
while ($i < 10)
{
  ...
  }
continue
{
  $i++;
}
```

Operators

TABLE 78. Perl Operators Fields

Operator	Description
!	Logical NOT operator.
||	Logical OR operator.
&&	Logical AND operator.
!=	Not equal.
#	Comment text.
==	Numeric equal.
<	Numeric less than.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 78. Perl Operators Fields (Continued)

Operator	Description
>	Numeric greater than.
<=	Numeric less than or equal to.
>=	Numeric greater than or equal to.
++	Auto-increment operator.
--	Auto-decrement operator.
<=>	Numeric comparison, returns **-1**, **0**, or **1**.
=~	Certain operations search or modify the **$_** string by default. This operator makes that kind of operation work on some other string. • The *right* argument specifies a search pattern, substitution, or translation. • The *left* argument specifies what to search, substitute, or translate instead of the default **$_**. • The *return value* indicates the success of the operation. If the *right* argument is an expression other than a search pattern, substitution, or translation, then when you run the script, Perl interprets this argument as a search pattern. This is less efficient than an explicit search, because Perl must compile the pattern every time it evaluates the expression. The precedence of this operator is lower than *unary minus*, *auto increment*, or *auto decrement*, but higher than everything else.
!~	Similar to =~ (except that Perl negates the return value).
**	Exponentiation operator.
**=	Exponentiation assignment operator.
()	Null list, used to initialize an array to null.
.	Concatenation of two strings.
.=	Concatenation assignment operator.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 78. Perl Operators Fields (Continued)

Operator	Description

x	Repetition operator. Returns a string, consisting of the *left* operand, repeated the number of times specified in the *right* operand.
	In an array context, if the left operand is a list in parentheses, this operator repeats the list.
	`print '-' x 80;# print row of dashes`
	`print '-' x80;# illegal, x80 is identifier`
	`print "\t" x ($tab/8), ' ' x ($tab%8);`
	`# tab over`
	`@ones = (1) x 80;  # an array of 80 1's`
	`@ones = (5) x @ones;# set all elements to 5`

x=	Repetition assignment operator. Works only on scalars.
..	Range operator. This is really two different operators, depending on the context: In an array context, returns an array of values, counting (by ones) from the left value to the right value. You can use this context to write **for (1..10)** loops, and to run slice operations on arrays.
eq	String equality (== is numeric equality). For a mnemonic, think of **eq** *as* a string, that *represents* an equality string. In Perl, you no longer have to compensate for the awkward and traditional behavior of using == for *both* string *and* numeric equality.
le	String less than or equal to.
ge	String greater than or equal to. l
cmp	String comparison. Returns **-1**, **0**, or **1**.
ne	String inequality (!= is numeric inequality).
lt	String less than.
gt	String greater than.

Functions

This section describes **Functions**, which are Veritools API commands that allow Perl to communicate with the Undertow waveform window.

You can use a **Function** call as a stand-alone command, which discards the return value. Or you can use the return value of a function call as a string or list of strings. The syntax of a function call is:

function_name(expr1, expr2, ...).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Table 79, “Perl Functions Fields” describes the available functions.

TABLE 79. Perl Functions Fields

Function Name	Description
CloseAllFilesAndOpen(*filename*)	Closes all dumpfiles, and opens the dumpfile with the specified *filename*.

CompareLogicSignals(*signal_name1*, *signal_name2*, <*min_time*>, <*max_time*>)	Returns a list of times when the two signals differ (The list is formatted as:
	`start_time1, end_time1,`
	`start_time2, end_time2`
	and so on.

CreateAnalogSignal(*signal_name*, *expression*)	Creates an analog signal with the specified name, using the specified analog functions expression. Enclose expression strings in double quotes. For example: `CreatAnalogSym($sig1,”(log(abs ($sig2)))”);`
CreateNewVariable(*variable_name*, *variable_type*)	Creates a new variable with the specified name and type, where *type* is one of the following: • ANALOG (time x, floating point y) • ARBITRARY (floating point x, floating point y) • LOGIC <*size in bits*> For example, **LOGIC8** is an 8-bit logic variable, time x, binary data.
CreateXAxis(*x_axis_index*, *x_axis_type*)	Creates a new X axis (pane) of the specified type, at the specified index position. Moves any existing panes down one index position, to make room, as necessary. Valid values for the x axis type are: • time • arbitrary *Note:* ***x_axis_index*** *starts at* ***1****. If you use an x_axis_index of* ***0****, the new X axis (pane) becomes the* ***last*** *X axis (pane).*
CutSignals()	Cuts all selected signals from the current X axis (pane).
DeselectAllSignals()	Deselects all signals in the current X axis (pane).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 79. Perl Functions Fields (Continued)

Function Name	Description
DisplayVariableAddTo Overlay(*variable_name*)	Adds the specified variable to the display window.
EndTime()	Returns the maximum time of the current pane.
FindSignals(*module_name*, *variable_regular_expression*, <*check_submodule*=1>)	Returns a list of strings, containing the full hierarchical names of *all* variables in the specified module, whose names match the regular expression. Also lists all sub-module names that match the regular expression, if the option is non-zero.
GetCurrentXAxisIndex()	Returns the *x_axis_index* for the currently selected X axis (pane). *Note:* ***x_axis_index*** *starts at* ***1***.
GetCursor(*cursor_number*)	Enter a cursor number as an argument (**0**, **1**, or **2**). This function returns the time when that cursor is currently placed.
GetSelectedSignals()	Returns a list of strings, containing the names of the signals that are currently selected in the display window.
GetSelectedYAxisIndex()	Returns the *y_axis_index* for the first selected Y axis (overlay) in the current X axis (pane). *Note:* ***y_axis_index*** *starts at* ***1****, and returns* ***0*** *if no Y axis is selected.*
GetValueAtTime(*signal_name, time*)	Returns the value of the specified signal, at the specified time.
GetXAxisCount()	Returns the number of the X axis (panes) in the display window.
GetXAxisTitle(*x_axis_index*)	Returns the title of the specified X axis (pane).
GetXAxisType(*x_axis_index*)	Returns the *type* of the specified X axis (pane). Valid types are: • time • arbitrary *Note:* ***x_axis_index*** *starts at* ***1***.
GetXAxisZoom()	Returns an array, containing the displayed start and end **X** values, for the current X axis (pane).
GetYAxisCount()	Returns the number of Y axis (overlays), in the current X axis (pane).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 79. Perl Functions Fields (Continued)

Function Name	Description
GetYAxisSignalCount(*y_axis_index*)	Returns the number of signals in the specified Y axis (overlay), in the currently-selected X axis (pane). *Note:* ***y_axis_index*** *starts at* ***1***.
GetYAxisZoom()	Returns an array, containing the displayed minimum and maximum Y values for the first selected Y axis (overlay), in the current X axis (pane).
GoToValue(*signal_name, regular_expression*)	Advances the T0/X0 cursor to the time/X, where the value of the specified signal matches the specified regular expression. Returns **1** on success, or **0** on failure. *Note: Do not use a regular expression for an analog value. Use the numerical value to match.*
MainLoop(*widget_name*)	Use this routine, instead of **TKMainLoop**. Otherwise, the main Undertow windows do not respond (To receive events, you must pass a string containing the name of a widget).
MaxTimeX()	
MinTimeX()	
ModuleGetModules(*module_name*)	Returns a list of strings, containing the names of all sub-modules for the specified module.
ModuleGetVariables(*module_name*)	Returns a list of strings, containing the names of the variables for the specified module.
OpenDumpFile(*dumpfile_name*)	Opens the specified dumpfile, so Undertow can read it. Returns the file ID string, to use for choosing signals.
OpenProjectFile(*filename*)	Opens the specified project file, so Undertow can read it. Returns **1** on success, or **0** on failure.
PlaceAnalogCursor(*cursor_number*, *y_value*)	Places the specified Y cursor, on the first selected signals in the current X axis (pane), at the specified Y value.
PlaceCursor(*cursor_num, time/X*)	When you enter a cursor number (**0**, **1**, or **2**) and a time as arguments, this function places the specified cursor, at the specified time. You can use a cursor number of **3**, to place a *grid* cursor. Returns an empty list of strings.
PlaceTagUsingCursors(*tag_text*)	Places a tag on the selected signal, using the X0/Y0 and X1/Y1 cursors as the anchors, with the specified tag text. Returns nothing on success, or an error string on failure.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 79. Perl Functions Fields (Continued)

Function Name	Description
PlaceTitleTag *y_axis_index*, *tag_text*()	Places a tag on the specified Y axis (overlay), using the specified tag text. *Note:* ***y_axis_index*** *starts at* ***1***.
PlaceValueTagUsingCursor(*tag_text*)	Places a value tag on the selected signal, using the X0/ Y0 cursor as the anchor, with the specified tag text. Returns nothing on success, or an error string on failure.
PrintSelectedSignalsToFile(*filename*, *<start_time/x>*, *<end_time/x>*, *<time/x per page>*)	Prints the selected signals (or all signals in the pane if no signals are selected) to the specified *filename*. Optional parameters are: • Starting x/time to print. • Ending time/X to print. • Amount of time/X to print on each page. These optional parameters default to printing the currently-displayed interval on the pane, onto a single piece of paper. Use the **Print Signals** dialog box to specify other print options.
RemoveXAxis()	Removes the current X axis (pane) from the display window.
Restore(*filename*)	Restores the *savefile* that the *filename* specifies.
SelectSignal(*y_axis_index*, *<y_axis_signal_index>*)	Selects the signal in the specified Y axis (overlay) of the current X axis (pane), at the specified signal index. If you omit *y_axis_signal_index*, then this function selects all of the signals in the Y axis (overlay). *Note:* ***y_axis_index*** *and* ***y_axis_signal_index*** *both start at 1.*
SelectXAxis(*x_axis_index*)	Selects the specified X axis (pane), to use as the current X axis (pane). *Note:* ***x_axis_index*** *starts at* ***1****. A value of* ***0*** *selects the bottom X axis (pane).*
SetXAxisTitle(*x_axis_index*, *title*)	Sets the title for the specified X axis (pane). *Note:* ***x_axis_index*** *starts at* ***1***.
SetYAxisZoom(*minimum_y*, *maximum_y*)	Sets the minimum and maximum display Y values, for all selected signals in the current X axis (pane). *Note: To use the default zoom, set both the* ***minimum_y*** *and* ***maximum_y*** *to* ***0****.*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 79. Perl Functions Fields (Continued)

Function Name	Description
SortOutput()	Sorts the contents of the Output window. This function performs the same operation as clicking the **Sort** button in the Output window. This function returns an empty list of strings.
TestAanalyzerSetTimingSignal(*signal_name*, {“LE”, “TE”}, *minus_time*, *plus_time*)	Sets the timing clock information, which you normally set in the **Test Analyzer** window. Undertow uses this information when comparing signals.
VariableGetIncrementalValue()	Initializes the incremental data access routine, for the specified variable. This function does not return a value.
VariableGetType(*variable_name*)	Returns a string containing the type, for the variable with the specified name. Possible types are: • ANALOG (*time X*, *floating point y*) • ARBITRARY (*floating point X*, *floating point y*) • EVENT (*time x*, *binary data*) • LOGIC <*size in bits*> (for example, **LOGIC8** is an 8-bit logic variable, *time x*, *binary data*).
VariableGetValues(*variable_name*)	Returns a list containing the data for the specified variable. The list consists of an alternating sequence of a floating point *time/x*, and a *value* string.
VariableSetValues(*variable_name*, *array_of_x1_y1_x2_y2_etc*)	Uses the data values, for the variable with the specified name, as the specified array of data. The array of data consists of alternating elements of *x1*, *y1*, *x2*, *y2*, and so on.
Zoom(*start_time/x*, *end_time/x*)	Zooms the current pane, to the specified starting and ending values.

Example Script

```
#
  # This script counts the number of transitions
        # from low to high for the selected signals, and
        # displays the results for each signal.
        #

        print "\nNumber of transitions from '0' to '1':\n";
        @sigs = GetSelectedSignals();
        $num = @sigs;
        for ($i = 0; $i < $num; $i++) {
            PlaceCursor(0, 0);
```

```
        for ($count = 0; 1; $count++) {
        $status = GoToValue(@sigs[$i], "0");
        if (! $status) { last; }
                $status = GoToValue(@sigs[$i], "1");
            if (! $status) { last; }
    }
  print @sigs[$i], ":  ", $count, "\n";
}
```

1. Use the Script Tool to enter the code.
2. To run the script, click **Apply**.

   A separate window shows the output. This window has the same button as the script tool.
3. To avoid confusion, clear this output window before you run another script.

   If your Perl script has errors, then an **Error** window opens when you click **Apply**.

   The **Error** window shows:

   - A list of errors.
   - The line number containing each error.
   - A suggestion of what might have caused the error.
4. To avoid confusion, clear this error window before you run another script.
5. After you complete your Perl script, you can bind a key to that script.

   For example, to attach a bind key to the *place_cursor* script, add the following lines into your **.Xdefaults** file:

   ```
   UNdertow_Display*display_time_clip.translations: #overide\n\
   Ctrl<key>X:   commandRunScript($DESIGN/config/scriptDir/bin/sun4v/place_cursor.script)\n\
   ```
6. After you add the UNdertow_Displayer line once, you can just list your bind keys under that line.

Builder, a graphical Perl window building tool

The script tool not only builds textual Perl scripts that can interact with Undertow, but also includes a graphical window builder for Perl, called *Builder*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

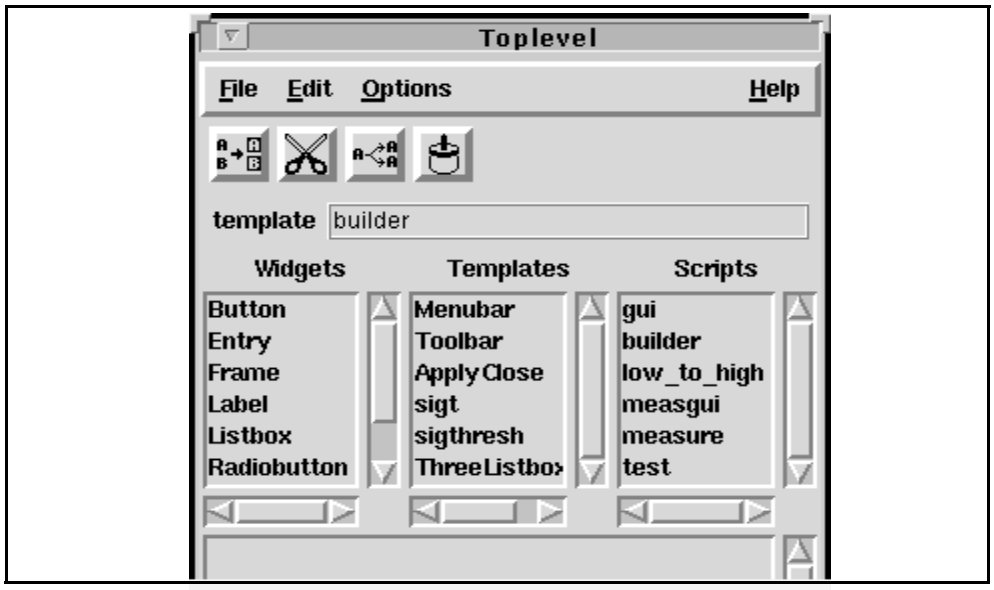
Note: To ensure that scripts run properly, place the images *subdirectory under* scripts*.*

You can use the script tool to write scripts, and then use *Builder* to build a graphical window that facilitates the execution of this script. This graphical window can contain:

- Text areas for you to enter input.
- Text areas for Undertow to output the results.
- Buttons, onto which you can attach scripts.
- Other graphical elements to make running your scripts easier.

You can use Builder to create a *template* (a partial graphical window), and save it in the **Templates** window. You can then use the template as a basis for building other windows, which simplifies the building of windows. You can build templates, and then use them repeatedly to create many different windows.

FIGURE 152. Builder Window



The Builder window, as shown in Figure 152, “Builder Window”, has three sections:

- Widgets.
- Templates.
- Scripts.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Widgets

The **widgets** window contains low-level widgets that you can use to build templates or windows. When you select any widget, Undertow inserts it onto a window frame. Use the middle mouse button to reposition or move the widget.

These widgets are:

- **Buttons**—Pops up the **Button** widget. This widget adds push buttons to the window.
- **Entry**—Pops up an **Entry** widget. Use entry widgets to type text, drag and drop text, or display text results.
- **Frame**—Serves as geometric frame. The separator is made from the frame widget, but is changed into a thick line.
- **Label**—A text label that you can use to label areas in the window.
- **List Box**—This widget can hold a text list.
- **Radio Button**—This widget holds a button and a text label.
- **Scroll Bar**—Pops up the scroll bar widget.
- **Text**—Pops up the text widget, which you can use to enter or display text.

  After you place a widget onto a window, you can move it to any position. When you press the right mouse button over a widget, you can select characteristics attached to the selected widget. The following sections describe these widget characteristics.

Command

This option opens a window displaying the current command that is attached to this widget. To attach a new command to the widget:

1. Enter a command in the **New Command** window.
2. Click Apply.
3. Click Close.

The new command attaches to this widget. Commands are generally names of Perl subroutines. To attach a Perl script, instead of a subroutine, enter the following in the **New Command Window**:

```
&DoScript(“Script_name”);
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

NOTE: The &DoScript("Script_name"); causes the script_name to be executed.

Configure

This option opens the configuration for this widget. The configuration for the widget is displayed, and windows show both the current value and the new value for the configuration parameter.

1. Press on any parameter in the right list box to see its current value.
2. To change the size of a widget, press the **Shift** key, and then press the middle mouse button on any widget.
3. To stretch or shrink the size of this widget, drag the mouse button.

Destroy

This option removes the widget.

Flash

This option flashes a specific widget, so you know that the cursor is over this particular widget.

Text

You can use this option to view the **Current Text** attached to this widget. You can also use the **New Text** area to attach new text. For example, the button widget opens with the default text label, **A Button**, which you can then change to a text label that describes the action of this button.

Variables

All widgets that interact with a Perl script must have a variable name assigned to them, so that the Perl script can refer to this name. When you build templates, the widgets in the template can have a variable name. You can then change that name when you use this widget to create a new window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Templates

The Templates window holds partial windows that you build using widgets. You can then invoke this set of Widgets in building a complete window. To create templates:

1. Build a window.
2. To save that window as a template, use either:

   File => ***Save*** => ***Template***, or

   File => Save as => Template

Scripts

The Scripting window holds completed scripts. After you build a window, if you save it as a script, it appears here.

File => ***Save*** => ***Script***

File => ***Save as*** => ***Script***

Steps in making scripts and windows for scripts.

To develop script, follow these basic steps:

1. Use the Scripttool to develop the part of the script that defines the interaction with Undertow:
   - Use hard-coded input.
   - Print the output in plain text.
   - Verify that the script works as expected.
2. Encapsulate the script in a subroutine, and save the subroutine.
3. To create the window (or GUI), use the **Builder** in Undertow.
   - Configure the **command** attribute of the **Apply** button, so it invokes the saved subroutine.
   - Verify that the output is still correct.
4. Remove the hard-coded input in the script.
   - Replace this input with data obtained from the GUI.
   - Verify that the results are as expected.
5. Send the output of the subroutine to the GUI, rather than printing it. Then verify correctness once again.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Examples of creating a simple script, and building windows.

To create a script, type it into the text area and save it. The following examples show you the steps to follow, to create some simple scripts.

Example #1: Creating A Simple Window

1. Open **Builder** from the scripts window.

   For example, **Window** => **Show Selector => Scripts** opens the script window**.**

2. When the script window opens, click on **Builder**.

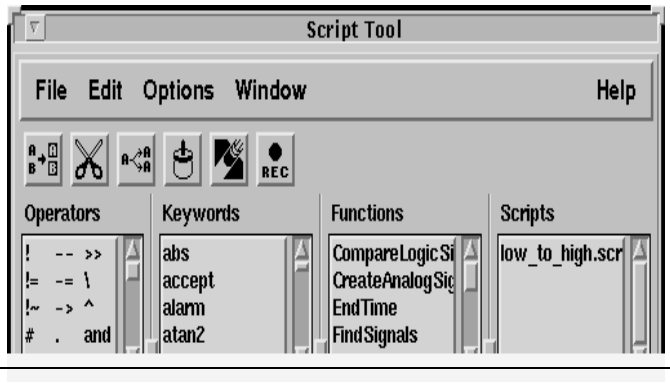


FIGURE 153. Script Tool Window

Creating a Label

1. Click on **Label** from the widgets list.

   The **Results** window opens, with a **Label** widget in the upper left corner. The label on the Label widget reads **A Label** by default.

2. Use the middle mouse button to drag the widget one centimeter down and one centimeter to the right.

3. Place the mouse over the new **Label** widget.

4. Using the right mouse button, click on the widget.

   A menu pops up.

5. Select **Text** from the popup menu.

   A dialog box titled **Text ?** pops up.

6. In the **Text ?** dialog, enter **Thing One:** under the words **New Text:**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

7. In the **Text ?** dialog, click **Apply**, and then click **Close**.
8. Click on **Label** from the widgets list.

   The new **Label** widget appears in the upper left corner again.
9. Drag the new **Label** widget below the **Thing One:** label.
10. Change the text for this second label to **Thing Two**.
11. Click on **Entry** from the widgets list.
12. Drag the Entry widget to the right of the **Thing One:** label.

Setting a Variable

1. Move the mouse over the new **Entry** widget.
2. Using the right mouse button, click on the widget.

   A menu pops up.
3. Select **Variable** from the popup menu.

   A dialog box titled **Variable ?** pops up.
4. In the **Variable ?** dialog, enter **thingOne** under the words **New Variable:**
5. Click on **Entry** from the widgets list.
6. Drag the new **Entry** widget to the right of the **Thing Two:** label.
7. Change the widget variable for this second entry to **thingTwo**.
8. In the **Variable ?** dialog, click **Apply**, and then click **Close**.
9. Click on **Button** from the widgets list.
10. Drag the new button widget until it is beneath the *labels* and *entry* fields.
11. Change the text of the button to read **Swap**.

Adding New Perl Code

1. Press the right mouse button.

   A menu pops up.
2. Choose **Command** from the popup menu.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

A dialog titled **Command ?** pops up.

3. In the **Command ?** dialog, enter the following lines of Perl code, underneath the words **New Command**:

```
$thingThree = $thingOne;
$thingOne = $thingTwo;
$thingTwo = $thingThree;
```

4. In the **Command ?** dialog, click **Apply**, and then click **Close**.
5. Click the left mouse button in the *entry* field for thing one, and enter **1111**.
6. Click the left mouse button in the *entry* field for thing two, and enter **2222**.
7. Click the **Swap** button.

Saving Your Script

1. In the menus, select ***Options => Script type***.

   The **Script Type** dialog pops up.

2. Set the script type to **script**.
3. In the menus, select ***File => Save As***.
4. The file selection box opens.
5. In the file dialog, enter **swap** in the entry at the bottom.
6. Click **Accept**.

Creating a Template

Templates are reusable script fragments. For example the commonly appearing pair of buttons, **Apply** and **Close**, are a perfect application for a template.

1. To start creating a template, either:
   - Click the **Select All** button**,** and then click the **Cut** button (the left-most) on the Builder toolbar, to clear all current window buttons and start with an empty template in the current Builder window, or,
   - Restart **Builder**, to open a new Builder window with an empty template.
2. Click on **Button** from the widgets list.

   Builder creates a new button.

3. Place the mouse over the new button.

4. While pressing the shift key, use the middle mouse button to drag the right edge over of the new button to the center of the results window.

   The button resizes as you drag it.

   As you hold the right mouse button, a menu pops up.

5. From the popup menu, change the text of the new button to **Apply**.

6. Click on **Button** from the widgets list.

7. Drag the new button to the right of the **Apply** button.

8. Resize the new button to match the **Apply** button.

9. Change the name of the new button to **Close**.

10. Using the menu that the right mouse button pops up, change the command of the **Close** button to:

```
DialogQuit ($top);
```

11. Reduce the height of the **Results** window until it is only slightly taller than the **Apply** and **Close** buttons.

12. Center the two new buttons vertically in the window.

The template is complete, except for creating the command behind the **Apply** button. What actions occur when you click **Apply**, depends on what the script that uses this template will do. So leave this command empty for now.

Saving a Template

To save the template, follow these steps:

1. In the Builder pulldown menus, select ***Options*** => ***Script type***.

   The **Script Type** dialog pops up.

2. In the **Script Type** dialog, set the script type to **template**.

3. In the Builder pulldown menus, select ***File*** => ***Save As***.

   The file selection dialog box opens.

4. In the file selection box, enter **apply_close** at the bottom, and click **Accept**.

   Note: Do not use dots in the names of scripts or templates.

   When the script template has been saved, and is ready to be imported into a script, it appears in the templates list.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Example #2

1. Start the **Scripttool** from Undertow.
2. Click in the text area.
3. Enter the following text:

   ```
   print "hello\n";
   ```

4. Click **Apply**.

   Your script prints hello to the results window in the script tool. Script tool always sends the output of the scripts it runs.

5. Start **Builder** (which is itself a Perl script) from the Scripttools, as described below.

Starting Builder

1. If the **Scripts** list-box is not visible in the Scripttool, select ***Windows => Show Selector*** from the pulldown menus.
2. Click on **builder** in the **Scripts** list-box of the Scripttool.
3. Click **Apply**.

   The **Builder** window opens:

4. Click in the text area in **Builder**.
5. Enter the following line and press **Return**:

   ```
   print "hello\n";
   ```

6. Click the **Apply** button in the Builder.

   The output of your second script, again the word hello, again appears in the Scripttool results window. This occurs because Scripttool is running **Builder**, and Builder is running your second script. But unlike Scripttool, Builder does not create a results window for the output of the scripts it runs. Instead, it prints that output as if it were its own.

   Builder *does* include a results window, but uses that window to display the *GUI* output of the script it builds, rather than the *textual* output. The results window is the big, empty, square window that pops up when you click **Apply** in Builder. It is empty because, so far, your script in Builder does not create any graphical components; it contains only a **print** statement.

Adding A Label

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1. In the **Templates** list-box, click on **Menubar**.

   Builder places the menu bar into the upper left corner.

2. In the **Templates** list-box, click on **Toolbar**.

   Builder places the toolbar directly underneath the menu bar.

3. In the **Widgets** list-box, click on **Label**.

   A label widget with the words **A Label** appears in the upper left corner of **Builder**.

   Note: ***Builder*** *does not automatically decide where to place any other objects that you create, nor does it know yet what text to place in the label.*

4. Using the middle mouse button, drag the label until it is directly underneath the toolbar, and then move it slightly to the right.

5. To configure the text on the label, press the right mouse button over the label, and select **Text** from the pop-up menu.

   A dialog opens, asking you to enter the new text that will appear.

6. Enter **template** in the entry field, underneath the words **New Text:**

7. Click **Apply**, and then click **Close**.

Adding List Boxes

1. In the **Widgets** list-box, click **Entry**.

   An entry widget appears in the upper left corner of Builder.

2. Drag the entry widget until it is immediately to the right of the new label.

3. In the **Templates** list-box, click on **Threelist-boxes**.

   Builder places a frame containing three list-boxes, without scrollbars, in the upper left corner of the screen.

4. Turn highlighting on before attempting to drag the frame.

5. In the pulldown menus, select ***Debug*** **=>** ***Highlight***.

6. Slide the mouse across the GUI that you are creating.

   Builder shows the widgets in reverse video while the mouse is over them, so you can see what you are about to drag.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

7. Drag the frame containing the list-boxes straight down, until it is beneath the label and entry field.

   Note: Leave room for one more row of labels between the label and entry field, and the list-boxes.

Labeling the List Boxes

1. In the **Widgets** list-box in Builder, click three times on **Label**.

   Builder adds three labels, one on top of another, in the upper left corner of the **results** window.

2. Drag each label to a point just above each of the three waistbands.

3. Slide the mouse over the first (left-most) label.

4. Press the right mouse button, and select **Configure** from the popup menu.

   The configuration dialog opens. Two list-boxes appear:

   - The left box is filled with all of the existing widgets in the **Builder** program.
   - The right box is filled with all of the configurable attributes of the widget that you are configuring.

5. Scroll to the bottom of this list, and click on **text**.

   The words **A Label** appear in the **Current Value** text widget.

6. Click in the **New Value:** text area, enter **Widgets**, and click **Apply**.

7. Slide the mouse button over the second (middle) of the three labels that you created.

8. Refresh the configuration dialog the same way you opened it the first time, using the right-mouse-button popup menu.

9. Scroll down the right list-box, choose **text**, and change the text of this label to **templates**. Also change the text in the right-most label to **Scripts**.

10. Click **ApplyClose** in the templates list.

Creating and Activating the Window

1. Drag the frame containing the **Apply** and **Close** buttons, until it is just below the text area, and center it in the window.

   Note: This step is difficult, because there is not much room to grab the frame.

2. Resize the window, until the height fits the graphical elements in the window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

*Note: You are creating a duplicate of the **Builder** window!*

3. Attach the **Apply** button to a script that does what you want.
4. Attach the other widgets of your new GUI to provide input to that script, and also to display the results.

Example #3: Script to measure analog signals

The following is an example of creating a script, and using the Veritools **Builder** to build a window that you can use to execute this script.

The following script, named measure.script, was designed to do the following:

- Read the names of two analog signals selected on the **Waveform** display.
- Read the values of **T0** and **T1**, the time cursors that define the outer bound, measuring between two thresholds on the two signals.
- Place the **T1** cursor on a user-defined threshold of the first signal.
- Place the **T0** cursor on a user-defined threshold of the second signal.
- Measure the difference between **T0** and **T1**, and place the result in the results output location.

```
$cursor1time = GetCursor (0);
$start = $cursor1time;
$start2 = $cursor1time;
print "cursor1time = $cursor1time \n";


$cursor2time = GetCursor (1);
$end = $cursor2time;
$end2 = $cursor2time;
print "cursor2time = $cursor2time \n";


&PlaceCursor(1, $cusror1time);
($signal, $signal2) = &GetSelectedSignals();
print "signal = $signal, signal2 = $signal2 \n";
if (not &GoToValue ($signal2, $threshold2)) {
  print "No value found on $signal2 \n";
}
$time2 = &GetCursor (0);
print "time2 = $time2 \n";


&PlaceCursor (0, $cursor1time);
print "just placed cursor 0 at $cursor1time \n";
if (not &GoToValue ($signal, $threshold)) {
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
  print "No value found on $signal \n";
}
$time1 = &GetCursor (0);
print "cursor 0 set on $signal set at $threshold at time= $time1\n";
&PlaceCursor (1, $time2);
print "cursor 1 set on $signal2 set at $threshold2 at time= $time2\n";
$measure = &GetCursor(1) - &GetCursor(0);
print "measure: " . $measure . "\n";
$results = $measure;
```

You can now build a window for this script, that will do the following:

- Hold the names of the two selected signals in two text areas, called **Signal**.
- Provide two threshold text areas, where you can enter the values of the thresholds:
  - The upper threshold area is for the first signal.
  - The bottom threshold area is for the second signal.
- Provide a **Start** and **End** text area for the boundary of this measurement, as defined by the locations of **T0** and **T1**.
- Provide a result area for the **T0** to **T1** measurements.
- Provide **Apply** and **Close** buttons, to control the operation of this window.

Creating Templates

You can use the **Builder** to build a window. In the Script tool:

1. In the pulldown menus, select ***Window => Show Selection => Scripts***.
2. In the Scripts list, click on **Builder**.
3. Click **Apply** at the bottom of the Script Tool.

   The Builder window opens.

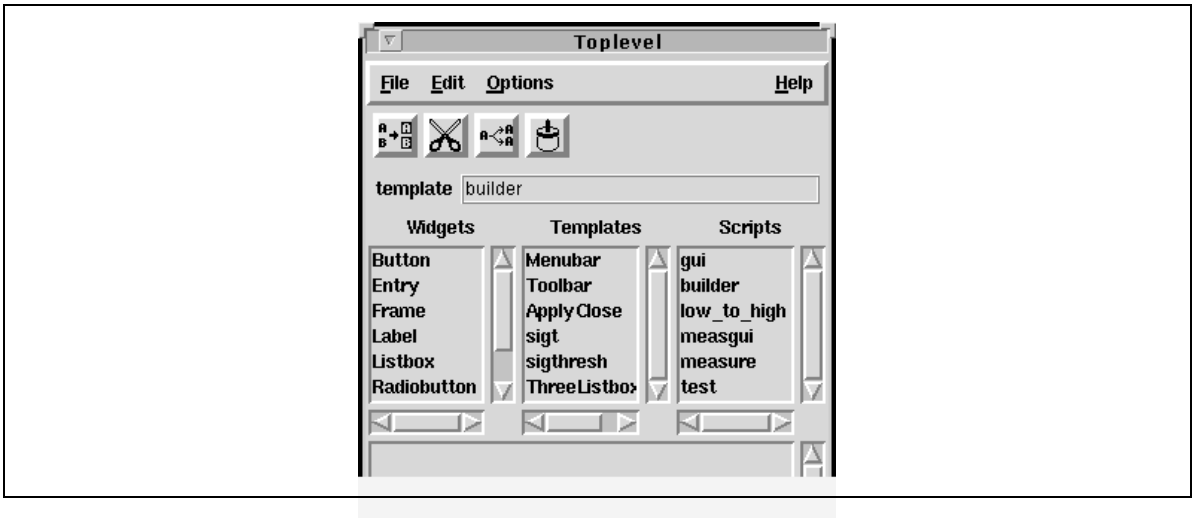


FIGURE 154. Script Tool

4. In the pulldown menus of the **Builder** window, select ***Templates*** => ***Menubar***.

   A menu bar opens in a window.

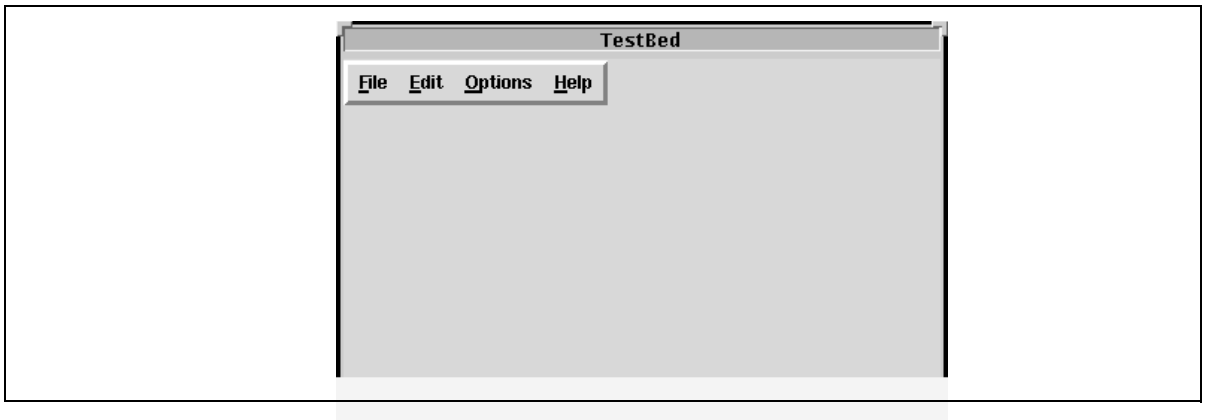


FIGURE 155. Menubar in Builder Window

5. In the pulldown menus of the **Builder** window, select ***Templates*** => ***Toolbar***.

   A Toolbar appears below the window in the frame.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

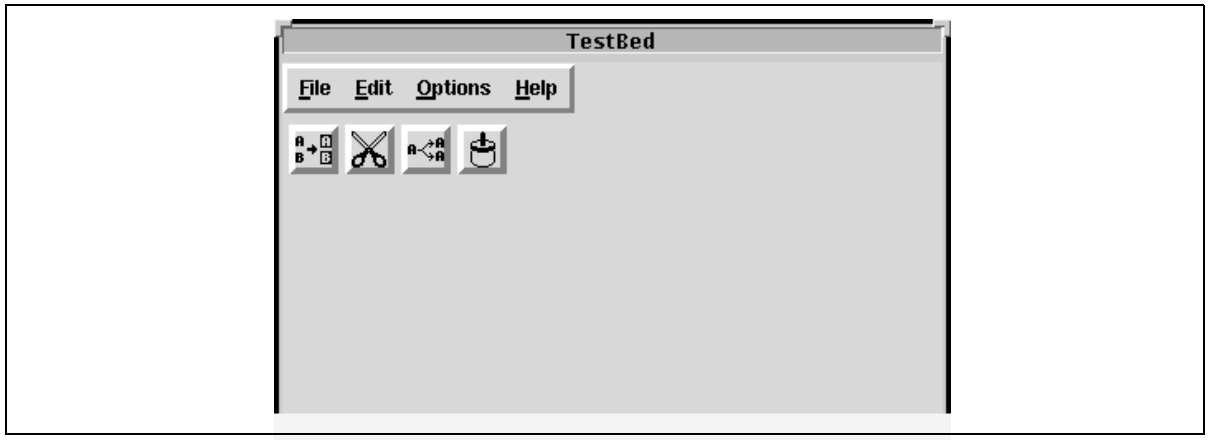


FIGURE 156. Menubar and Toolbar in Builder Window

6. In the pulldown menus of the **Builder** window, select ***Templates*** => ***sigt***.

   A template opens, with entries for signal, threshold, and start and end times to bracket the threshold.

7. Move this template down, to position it properly beneath the toolbar.

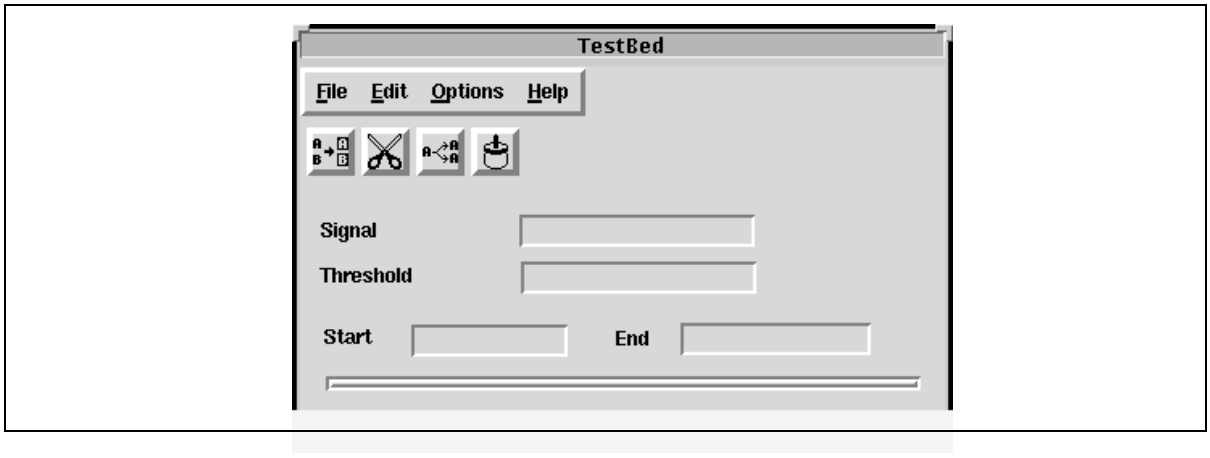


FIGURE 157. Template in Builder Window

8. In the pulldown menus of the **Builder** window, select ***Templates*** => ***sigthresh***.

   Another template opens, similar to **sigt**. However, this template also contains an **Update** button, which you can use to move the **T0** and **T1** cursors. You can then bracket the threshold again on signal 2.

9. Drag this new template down, until it is just beneath the **sigt** template.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

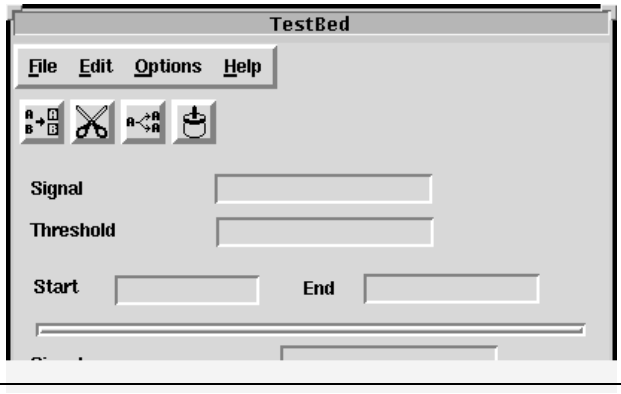


FIGURE 158. Second Template in Builder Window

Adding Widgets

1. In the widgets list, click **Entry**.
2. Reposition the entry to just below the templates.
3. Place the mouse cursor over this entry widget.
4. Use the right mouse button to open the popup menu, and select **Variable ?**

   The dialog titled **Variable ?** opens.
5. When the dialog opens, enter **result** in the entry field for **New Variable:**
6. Click **Apply**, then click **Close**.

Adding Labels

1. In the widgets list. click on **Label**.
2. Drag the new label to the left of the entry widget that you just added.
3. Using the popup menu that appears when you press the right mouse button, select **Text**.
4. Change the text of the new label to read **Result:**
5. In the templates list, click on **ApplyClose**.

   A template containing **Apply** and **Close** buttons appears in the upper left corner.
6. Drag the template down to the bottom of the window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Whenever you use a template in a script, Builder places the contents of the template in a frame (in this example, the frame contains both the **Apply** and **Close** buttons).

*Note: You must drag the entire frame, not just the **Apply** button or just the **Close** button. To do this, place the mouse cursor in the space between the two buttons, before you drag the frame.*

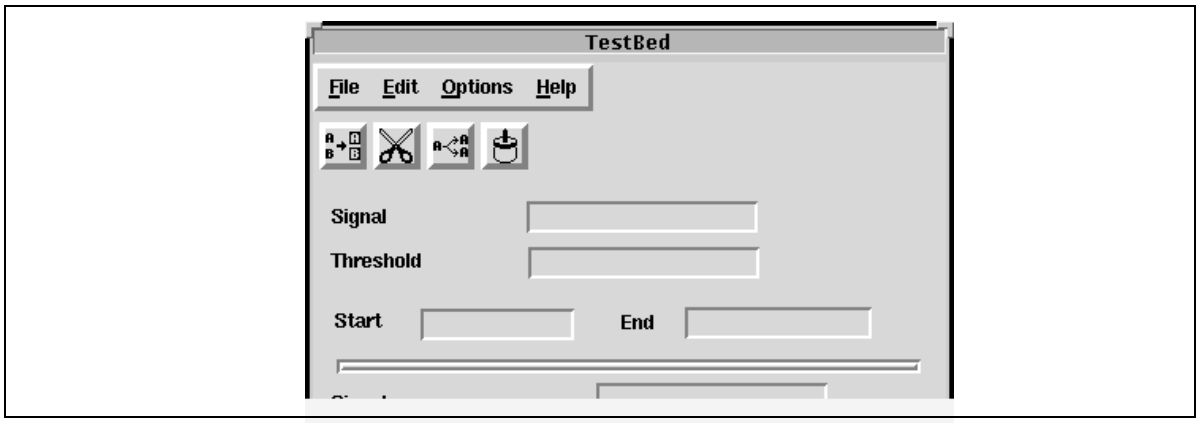


FIGURE 159. Completed Template in Builder Window

Attaching Scripts to Buttons in the Template

1. To set the command behind the **Apply** button, again press the right mouse button on the **Apply** button.
2. When the dialog titled **Command ?** opens, enter the following in the area labeled **New Command:**

   &DoScript(“measure”);

   Builder attaches the measures.script script to the **Apply** button. That is, when you click **Apply**, the measures script executes.
3. In the command window, click **Apply**, and then click **Close**.

   Your script is ready to save.

Saving Your Script

1. From the pulldown menus, select ***Options => Script Type***.
2. Set the script type to **script**.
3. From the pulldown menus, select ***File => Save As***.
4. To save the script under a new name, select the script name from the **scripts** list.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

In this example, click on the name **measuregui**.

5. Click **Apply** again.

   When you click **Apply**, the **measuregui** script runs, and invokes the **measures** script.

6. Read the **measures** script, to understand what it is doing.
7. Place your **T0** and **T1** cursors.
8. Enter the appropriate values in the **threshold** entry fields.
9. Click **Apply**, to view the results of the **measure** script.

Undertow Analog Extensions

This section explains how to use extension analog scripts to measure, analyze, and process analog signals.

Downloading Scripts

These Perl scripts use special routines, defined in special Perl modules:

UndertowAnalogExtension.pm
UndertowPerlUtilities.pm.

These Perl modules are part of regular Undertow releases from version 8.3.3. However, if the installed version is earlier than that, or if these additional modules are missing from the Undertow install directory:

1. Go to the Veritools web site (scripts_pool -> analog).
2. Download the two archive files, **analog_scripts.tar** and **analog_perl_modules.tar**. to the regular **scripts** directory of Undertow:

   <undertow_install_dir>/scripts

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Extracting Scripts

1. Use **tar -xvf** to extract the **analog_scripts.tar** and **analog_perl_modules.tar** files into the Undertow **scripts** directory.
2. Ensure that all of the required scripts are in this directory.

Scripts using Perl Analog Extension

The following scripts demonstrate the use of these functions.

*Note: Unless otherwise stated, range refers to that part of the signal that is bound by the **T0/Y0** and **T1/Y1** cursors. You can set these cursors. Default values are the **start** of the signal and the **end** of the signal.*

- **3DB.script**

  Computes 3DB points (up or down) from the maximum point in the selected range.

- **Amplitude.script**

  Finds the *amplitude* for the selected range. Amplitude is the absolute difference between the top line and the base line.

- **Average.script**

  Finds the average of **Y** values of the signal in the selected range.

- **AtX.script**

  Places a tag with **X** and **Y** values, at the point where the **X0/Y0** cursor is located.

- **Baseline.script**

  Finds the baseline for the specified signal, in the selected range.

- **DeltaX.script**

  Finds the difference between the cursor 1 time point, and the cursor 2 time point.

- **DeltaY.script**

  Finds the difference between signal values, at the time point of cursor 1 and the time point of cursor 2.

- **DutyCycle.script**

  Finds the duty cycle, which is defined as the pulse width divided by the period.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **MaxFall.script**

  Finds the maximum fall in signal values, in the specified duration.

- **MaxPoint.script**

  Finds the local maximum point.

- **MaxRise.script**

  Finds the maximum rise in signal values, in the specified duration.

- **MinPoint.script**

  Finds the local minimum point.

- **PeakToPeak.script**

  Finds the time difference between two top peaks, within the selected range.

- **ScriptMenu.script**

  Creates graphical menus, to invoke various analog scripts.

- **SlewRate.script**

  Computes the slew rate. This is the rate at which the maximum fall or the maximum rise occurs in the selected range.

- **Topline.script**

  Finds the topline for the specified signal, in the selected range.

Note: To see the working of these scripts, ensure that all of these scripts are in the ***<undertow_install_dir>/scripts*** *directory.*

UndertwoAnalogExtension.pm

The following functions are available in **UndertwoAnalogExtension.pm**:

- **sub Average($signal, $start_point, $end_point)**

  Computes and returns the average of signal values, in the specified start and end points.

- **sub FindFirstAndLastPeaks($signal, $start_point, $end_point)**

  Finds the first and last maximum and minimum peaks, for the specified signal, in the selected duration.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The returned value is an array of peak values, such as:

0: first maximum rise point
1: first minimum rise point
2: last maximum point
3: last minimum point

- **sub FindNDBPoints($signal, $start_point, $end_point, $ref_point, $dv_value, $precision)**

  Finds the *N* DB point (up or down if **-ve**) for the selected time point, within the selected range.

  Reference value used in the value of the selected signal, at the specified time point.

  Arguments are:

  - signal
  - start point of the range
  - end point of the range
  - reference time point
  - db value—For example, if you are looking for 3 db points, the value of this argument is **3** (the default value). For 3 db down, it is **-3**.
  - precision expected—The default value is zero, meaning that points are selected *only* if their value **exactly** equals the number that is *N* db (up or down) of the **ref_point** value.

  The returned value is an array of time points, where the value is *N* DB up or down of the value at the reference point.

- **sub FindNPeaks($signal, $start_point, $end_point, $no_of_peaks)**

  Finds the top *N* peaks (both maximum and minimum) for the specified signal, in the selected range.

  The returned value is an array of 4 hashes. Keys for all hashes are 1, 2, 3, ... N.

  1: Hash of top values
  2: Hash of corresponding top time points
  3: Hash of bottom values
  4: Hash of corresponding bottom time points.

- **sub FindPointsOfValue($signal, $start_point, $end_point, $value, $precision)**

  Finds the points that have signal value close enough (precision wise) to the specified value.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The returned value is the array of signal time points, whose values are near enough to the argument value.

- **sub GetAbsoluteValue($signal, $time_point)**

  Finds the absolute value of a time point, based on the array of values specified in **VariableGetValues**. To obtain this value, Undertow extrapolates the previous and next values, whenever the selected time point lies in between.

  However, the **GetValueAtTime** function returns values such as 272.43nA, for the actual value of 272.43343434-e07. The former value is of no use while comparing values of different time points.

  The returned value is the absolute value.

- **sub GetDutyCycle($signal, $start, $end)**

  Finds the duty cycle, which is defined as the pulse width divided by the period. The signal is expected to be periodic, with a clear positive or negative pulse.

  The returned value is the duty cycle.

- **sub GetFirstMaxPoint($signal, $start, $end)**

  Finds the first time point, where the value is the maximum in the selected range.

  The returned value is a time point, where the value is the maximum for the range.

- **sub GetFirstMinPoint($signal, $start, $end)**

  Finds the first time point, where the value is the minimum in the selected range.

  The returned value is a time point, where the value is the minimum for the range.

- **sub GetFirstSameValuePoints($signal, $start, $end, $precision, $percentage)**

  Identifies the first two time points of the signal that have the same value.

  The **$precision** argument determines which values Undertow considers as equal. If you do not specify a precision, Undertow uses zero by default; this means that a strict equivalence is required.

  The last argument, **$percentage**, indicates whether to apply the specified value as the number or as a %.

  - If the **$percentage** value is 1, it means interpret the precision value as a % change.
  - Undertow ignores any other value.

  The returned value is an array of three values:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

0: first time point
1: second time point
2: value

- **sub GetFrequency($signal, $start, $end, $precision)**

  Computes the frequency, which is 1/*period*.

  The returned value is the frequency number.

- sub GetMaxFall($signal, $start, $end)

  Finds the first maximum fall-by value, in the selected range.

  The returned value is an array of the following values, for the maximum fall:

  0: fall start point
  1: fall end point
  2: fall value

- **sub GetMaxRise($signal, $start, $end)**

  Finds the first maximum rise-by value, in the selected range.

  The returned value is an array of the following values, for the maximum rise:

  0: rise start point
  1: rise end point
  2: rise value

- **sub GetNextPointAfterCutoff($signal, $cutoff)**

  Finds the next point of an analog signal, after the cutoff.

  The returned value is either the next time point, or zero if the cut off point is invalid.

- **sub GetPeriod($signal, $start_point, $end_point, $precision)**

  Finds the period for the specified signal, in the selected time slice. The period is defined as the duration, after which the pattern repeats. The pattern is identified by the maximum rise and maximum fall.

  The returned value is the period number, or it can be an error message (if Undertow cannot find the period).

  The error typically regards an insufficient time slice—for example, not enough signals are available to recognize a pattern.

- sub GetPointOfValue($signal, $start, $end, $value)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Finds the next point of the specified value, after the specified starting point, and before the end point.

- The start point is excluded from checking, but the end point is included.
- If the start value is greater than the end value, then Undertow swaps both values.

The returned value is the time point (if the value is found) or **-1** (if no point is found).

- **sub GetPulseWidth($signal, $start, $end, $percent)**

  *Note: This subroutine finds only the points that the VariableGetIncrementalValue function generates. In that sense, it **does** work correctly when the point is to be extrapolated.*

  The returned value is the time point (if the value is found) or **-1** (if no point is found).

- **sub GetSanitizedSignalValues($signal)**

  Sanitizes the signal value array.

  - In some instances, **VariableGetValues** returns an array, but does not find the last pair, for the end time point. That is, the end point is later than the time point of the last pair.
  - If you *do* add the end point pair to the array, then the value of the end point is not available, and Undertow cannot extrapolate it. So it is same as that of the previous point.

  The returned value is an array of signal values, possibly one more pair added at the end.

- **sub GetSlewRate($signal, $start, $end, $type)**

  Computes the slew rate. This is the rate at which the maximum fall or the maximum rise occurs in the selected range.

  - This sub-routine uses either maximum fall or maximum rise, based on the argument.
  - The type is either the **FALL** or **RISE** word; Undertow considers anything else as fall.

  The returned value is the slew rate number.

- **sub GoToAtTime($cursor, $signal, $time)**

  This routine replaces three calls:

  - PlaceCursor
  - GetValueAtTime
  - PlaceAnalogCursor

  with one single call, to place the cursor at the specified value. This is similar to the **GoToValue** function, except that instead of the value it use the time.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This subroutine does not return a value.

- **sub PrintVarArray(@array)**

  This debug method prints an array. Arguments are in an array.

  This subroutine does not return a value.

- **sub VariableGetIncrementalValueAfterCutoff ($var, $cutoff)**

  Initializes the incremental data access routine for the specified variable, and for the subsequent incremental calls. It returns points that are after the cut off point. The next incremental point is immediately after the cut off.

  The returned value is the last point, where it exceeds or equals the cutoff.

UndertowPerlUtilities.pm

The **UndertowPerlUtilities.pm** module contains routines that the Undertow Perl routines use. These are *utility routines*, so they do not assume any knowledge about Undertow. These routines are generic, and are based purely on standard Perl, so they are independent of Undertow. Veritools created most of these routines because similar functionality was not available in either built-in routines in Perl, or the related **CSPAN** module.

The following functions are available in **UndertowPerlUtilities.pm**:

- **sub CompareAndInsertInHash(%exs_hash, $value, $limit, $com_type, $distinct)**

  Places the argument value in the ordered list of values in the hash. The Keys for this hash are the indices. In this case, the hash represents a sorted list of numeric values. The arguments are:

  - Existing hash.
  - Value to insert.
  - Limit on the number of values to maintain in the hash.
  - Comparison type:
    - Greater than 0, maintains top values and the list is descending.
    - Less than 0, maintains bottom values and the list is ascending.
  - Distinct value:
    - Greater that 0, maintains only distinct values and ignores any repeat value.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Otherwise, repeat values are not excluded.

  In effect, this indicates whether to use strict ascending / descending order, or non-strict order.

The returned value is a changed hash list. This list contains a pair:

- The key is **new_value_index**.
- The index of the new element as the value.

- **sub InsertInIndexedHash(%hash, $index, $value, $limit)**

  Inserts a new value at the indicated index. It pushes existing and subsequent values down in the index order. If the argument index is larger than the total count in the hash, it simply adds, using a count + 1 index value.

  Arguments are:

  - Hash.
  - Index where Undertow is to place the new value.
  - The value to place.
  - Limit:
    - If the limit is less than 0, no limit exists, and hash expands.
    - If the limit is 0 or larger, this method deletes all additional pairs.

  The returned value is a hash, which inserts the new value at the indicated position.

- **sub NDBValue($ref_value, $db_value)**

  Computes the value, for which the DB value is *N*. This value is typically 3DB (which is 3DB up) and 3DB Down.

  This function uses the following formula:

$$N = 20\log(E1/E2)$$

  where:

  - E1 is the reference value, and

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- E2 is value that this function computes.

Based on above formula, E2 is computed as:

$$E2 = \frac{E1}{10^{(N/20)}}$$

The arguments are:

- Reference value (numerator).
- DB value, *N*. If this value is **-ve**, it means **DB down**.

The returned value is a value (denominator), for which the DB value with a reference point is *N*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 9

Working With PLI

Overview

PLI (program language interface) makes it possible for the designers to go outside their Verilog design to other languages to create specific functions.

This chapter describes the syntax and semantics of the Veritools **PLI Tools**. These tools enhance simulation, by providing new system tasks.

- To use these tools, you must add some statements to your Verilog source files.

  This chapter describes various options that are available for these tools.

- The **PLI Tools** also require that you create a new simulator executable. That links in the vtpli.o file.

  The last section of this chapter describes how to create a new simulator executable, with a Veritools object file linked in.

Optimizing Tool

The **Optimizing Tool** is a Veritools **PLI** routine that outputs data from Verilog in a compressed form, which is typically a factor of 7 to 15 times smaller than the equivalent VCD data.

The data can be:

- signal data (replacing the VCD file), or
- connectivity data, or
- trace data.

You can also use the **Optimizing Tool** to create a batch file while you run the **Interactive Tool**. This way, you can release the Verilog license, while continuing to use the **Interactive Tool** to debug your source code.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The signal data file is organized so that you can quickly access data for a signal, even for large databases. Both the **Interactive Tool** and **Undertow** can process these files. You can also run Verilog in batch mode to create the needed files to completely debug source-level code, using the **Interactive Tool** and **Undertow**, without requiring a Verilog license.

Starting the Optimizing Tool

You can use the Optimizing Tool to dump data about connectivity and signal strength/transition. To do this, start the Optimizing Tool, using the **$vtDumpvars** system task, during a Verilog simulation. You can put the **$vtDumpvars** command in an initial block in your Verilog code, or you can enter this command directly at the Verilog prompt.

The **$vtDumpvars** system task works exactly like the Verilog **$dumpvars.**

- You can start the **$vtDumpvars** system task without any parameters, as follows:

  $vtDumpvars;

  This dumps all signals in your design.

- You can also start the **$vtDumpvars** system task with parameters that specify the signals to dump. These parameters are identical to the parameters that the Verilog built-in system task (**$dumpvars**) accepts**.**

This means that you can use a text editor to replace all occurrences of **$dumpvars** with **$vtDumpvars** in your Verilog source files.

$VtDumpvars Parameters

The first parameter must be an integer number. This number signifies the levels of hierarchy to apply to the remaining parameters.

Note: A zero for this parameter indicates ALL levels.

The remaining parameters can be either specific signals in your design, or specific instances of modules in your design. Again, this use the same principle as the standard **$dumpvars** in Verilog.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

For specific instances of modules, the first parameter determines how many levels down (starting from the specified instance) to dump signals. For example, your design might contain an instance of a CPU model, named top.CPU. You might want to dump ALL of the signals in this model, including model instances within the CPU model. To do this, invoke **$vtDumpvars** as follows:

$vtDumpvars(0,top.CPU);

However, for this same example, if you are interested in only the first three levels of the CPU model, then type:

$vtDumpvars(3,top.CPU);

You can invoke **$vtDumpvars** as many times as you want, and at any time during the simulation, to change the parameters that the **Optimizing Tool** uses.

As another example, your design might include the same CPU model as before, plus an instance of a PCI bus model at top.PCI. To dump the **phaseOneClk** signal from the CPU model, and all signals (four levels deep) in the PCI model, type:

$vtDumpvars(4,top.CPU.phaseOneClk,top.PCI);

Compression Modes

You can use the Optimizing Tool to trade off compression ratio for simulation performance.

- In the default mode, the **Optimizing Tool** achieves the best compression ratios, at the cost of simulation performance.
- However, if you select to dump only a *portion* of the complete design (such as the modules to debug), then the performance penalty is insignificant, even with optimal compression.

Table 80, “Compression Options” describes the flags that the **Optimizing Tool** passes to your Verilog command line. These flags provide much faster Verilog simulations, when you build files for use in Undertow.

TABLE 80. Compression Options

Mode	Description
+VTCOMPRESS20	**Default mode**(*For Undertow2004 and before*).
+VTCOMPRESS200	Real-time mode. Fast file identical to **vdump**, quick to create and to read, small file size. The file output for interactive mode is automatically set to this value

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

TABLE 80. Compression Options

Mode	Description
+VTCOMPRESS250	**Default mode**(*For Undertow2004 and up*). It compresses +VTCOMPRESS20 by 100X. For Highest Performance, add +VTVECTORVALUES, it forces all the bits data into vector values.
+VTCOMPRESS300	Fast file and all buses are vectored. Use this mode especially when you run Model tech.

Example

If:

- Your Verilog executable is named as099_ver, and
- You want to quickly create a file that is small and fast to read, and
- You want to view the file during real time,

then enter the following:

as099_ver -f verilogFile +VTCOMPRESS200 +VTVECTORVALUES

Dump File Names

By default, the **Optimizing Tool** dumps signal strength/transition data to the **vt.dump** file.

- To change this default, start the **$vtDumpnames** system task.
- This system task requires you to enter one argument: a string that specifies the name of the signal strength/transition dump file.
- You can start the **$vtDumpnames** system task only *once*.

*Note: The **$vtDumpnames** system task must precede the **$vtDumpvars** system task.*

Another way to specify the dump file name is to use the *plus* command line options. in Verilog Specifically, use the **+VTOPTZ_SFILE** *plus* command line option**,** to specify the name of the signal dump file.

For example, if you want to name the signal dump file myDumpfile, then you add the following to your Verilog command line:

+VTOPTZ_SFILEmyDumpfile

Note: Do not insert any spaces between the plus option and the file name.

Turning Dump File on/off

This command allows turning Dumping waveform file on/off at particular time.

Example:

```
initial begin
$vtDumpvars;
$vtDumpoff; //Start with dump off
#10000 $vtDumpon; //Turn dump on after 10000
#5000 $vtDumpoff; // Dump for 5000, then turn off
end
```

Trace Files

The **trace dump** file records each line of source executed as the simulator is running in a trace file. It is only required if you are using Undertow Suite to do source-code-level debugging, and you wish to use the virtual simulation feature. Virtual simulation is the ability to run Undertow Suite without checking out a simulator license, while having all of the apparent capability of running an active simulation.

Creating A Trace Dump File, Using Cadence Verilog-XL

If you use the Cadence Verilog-XL simulator, you can start the **vtrun** executable to create Trace dump files. Since the trace operation is built into Verilog-XL, the Veritools executable **vtrun** is used to pass to Verilog-XL the command **-trace**; and then to take the resulting trace information and reformat and compress it for use by Undertow Suite.

- When you start the **vtrun** executable, use all of the parameters that you would normally use to invoke the Verilog simulator.
- Additional parameters, specific to **vtrun**, control how you generate a **trace dump** file:

- The **vtrun -trace** command line argument generates a trace dump file, using the default file name, vt.trace.
- The **vtrun -tracefile *traceFileName*** command line argument specifies the name of the trace dump file, where *traceFileName* is the name that you want to assign to the trace dump file.

For a complete description of **vtrun**, refer to the **vtrun.ascii** or **vtrun.ps** file, which Veritools distributes with the **Optimizing Tool**.

The **vtrun** executable automatically enables collecting trace information, starting at time zero, whenever you specify the **-trace** command line argument.

However, collecting trace information can impact simulation performance. To control this, you can use the Verilog language constructs, at any time during the simulation:

- **$settrace;** enables collecting trace information.
- **$cleartrace;** disables collecting trace information.

For more information about selective tracing during a simulation, refer to the section describing the **$settrace;** and **$cleartrace;** system tasks, in the *Verilog Language Reference Manual*.

Creating A Trace Dump File, Using VCS and FinSim

If you use either the VCS or FinSim simulators, you can invoke system tasks from within your Verilog source code, to create trace dump files.

- To enable or disable tracing of your complete design, invoke the **$vtTrace** system task, as follows:

  $vtTrace(*enableFlag)*

  where *enableFlag* is either:

  - 1 to enable tracing, or
  - 0 to disable tracing.

- To enable or disable tracing of specified sections of your design, invoke the **$vtTrace** system task, as follows:

  $vtTrace(*enableFlag*,*levels*,root1,root2...);

  where

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- *enableFlag* is either:
  - 1 to enable tracing, or
  - 0 to disable tracing.
- *levels* is an integer indicating how many levels to apply the *enableFlag*. A zero for the *levels* argument signifies all levels.
- *root* arguments indicate which root(s) to enable or disable tracing, for the specified number of levels.

You can invoke the **$vtTrace** system task any number of times, with different parameters for each invocation.

To specify the name of the trace file, use the **$vtTracename(traceFileName)** system task. The *traceFileName* argument is a string that indicates the name of the trace file.

*Note: You must invoke the **$vtTracename** system task before the **$vtTrace** system task.*

PLI Dumping

General Descriptions

You can turn PLI dump on/off using $vtDumpon and $vtDumpoff commands to view waveform at particular time. This command allows users to control the $dumpvar on/off time without recompiling the design

Example:

```
initial begin
$vtDumpvars;
$vtDumpoff; // Start with dump off
#10000 $vtDumpon; // Turn dump on after 10000
#5000 $vtDumpoff; // Dump for 5000, then turn off
end
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Connection Analyzer

You can use the Undertow **Connection Analyzer** to traverse the structural connectivity of your design, to determine the cause of errors in circuit behavior. Undertow requires a *Connectivity* file for this function.You can use either of two methods to create a *Connectivity* file:

Use the $utConnectivity system task

This system task does not include any arguments.

- Place this system task in a Verilog initial block.
- Invoke this task only once per simulation.

This task produces a file named utConnectivity, which contains structural connectivity information about the design that you are simulating.

If you are running Undertow stand-alone, when you start the **Connection Analyzer** this way, a standard Motif file dialog prompts you for the name of the *Connectivity* file.

Select the utConnectivity file that corresponds to the file that you are viewing.

Use the Iv parser if you want to create an RTL schematic

1. Check the ***Options*** **=>** ***Monitors and Dumps*** radio button in **Iv**.
2. Check ***Use Project Name for Files***.
3. Fill in the **Project Name** field.
4. Click ***Simulator*** **=>** ***Run***, or ***Actions*** **=>** ***Load Batch Run***.

   The Connection Analyzer creates a connectivity file, whose name is the name of your project (*Project Name*) followed by the *connectivity* suffix.

   If you are running the Undertow Suite, the connectivity file as specified in Iv in ***Options =>Monitors and Dumps*** loads.

If you created the file through Iv and you are running the Undertow Suite, the **Connection Analyzer** loads the connectivity file, as specified in Iv in ***Options*** **=>** ***Monitors and Dumps***. The file filter is at your project directory path, and you need only to click on the name of the connectivity file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Creating a Memory Dump File (mem.dump)

- Use this option to dump the entire contents of memory to mem.dump:

  $vtDumpMem(*ramname*[0], 0, 255);

- Use this option during a write operation, to update mem.dump:

  $vtDumpMemAddr(*ramname*[0], *addr*);

How to View Memory Contents

1. To load the memory dump file from the **Waveform** window, select the ***File* => *Load Additional File*** menu item.

   This adds another *root* in the **Chooser** section of the **Source** window.

2. Open the hierarchy.

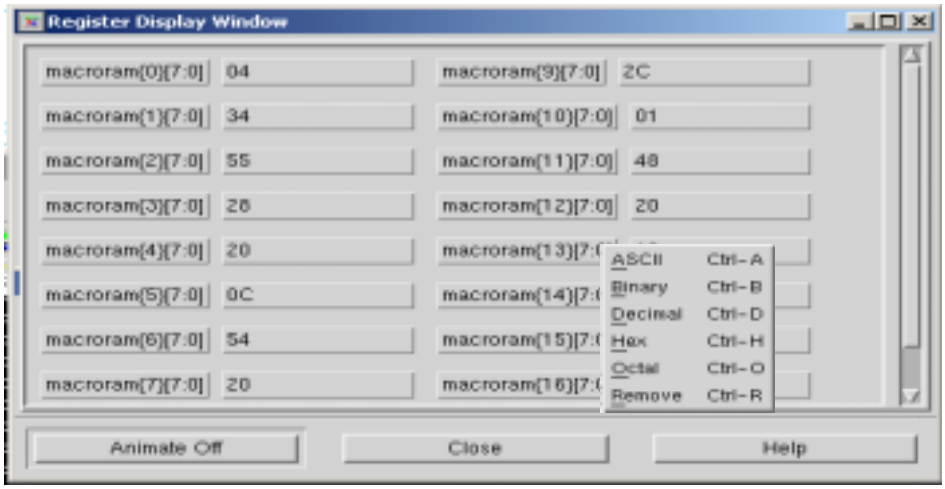
3. Display the memory, as you would in a signal.

   The **Waveform** viewer displays the memory addresses and memory contents. Users can also open Register Display Window to view the content.

Register Display Window

Click on options->Register Window to open up the Register Disply Window

FIGURE 160. Register Display Window



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

User can click to any part in the waveform and Register Display Window will automatically display the value of the regiester. User can also change the output format by left clicking on the register or:

1. ASCII (Ctrl-A)
2. Binary (Ctrl-B)
3. Decimal (Ctrl-D)
4. Hex (Ctrl-H)
5. Octal (Ctrl-O)

Or remove the Register Bit by clicking Remove (Ctrl-R)

Creating A Fast File On The Fly ($vtDump)

Use the **$vtDump** system task to create an Undertow *Fast File*, while simulation is running. This system task does not use any arguments.

$vtDump pipes the VCD file that Verilog creates, into the **vdump** executable. Veritools provides this executable with the PLI distribution. Therefore, the directory where the **vdump** executable resides, must be in your **PATH** environment variable.

After simulation completes, **$vtDump** generates a *Fast File*, but does not generate a VCD file. The fast file name is the same as what the VCD file would have been called, with a .fast extension added.

For this system task to operate correctly, you must execute this task *after* any calls to **$dumpfile,** and the call to **$dumpvars**. **$dumpfile** and **$dumpvars** are built-in systems tasks that are provided with Verilog, and are not part of the Veritools PLI tools.

*Note: If you are using the **$utLimitVCD** system task, you cannot use the **$vtDump** system task.*

Limiting A VCD File

You can use the **$utLimitVCD** system task to create a revolving VCD file of a fixed size. This system task requires one argument: the maximum size (in megabytes) of the VCD file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

For example:

$utLimitVCD(5);

limits the VCD file that Verilog generates, to include *only* the last five megabytes of simulation data.

For this system task to operate correctly, you must execute this task *after* any calls to **$dumpfile**, and *before* the call to **$dumpvars**. **$dumpfile** and **$dumpvars** are built-in systems tasks that are provided with Verilog, and are not part of the Veritools PLI tools.

*Note: If you are using the **$vtDump** system task, you cannot use the **$utLimitVCD** system task. This system task requires that the directory, where the **utdmp** executable resides, must be in your **PATH** environment variable. This executable is shipped with the Veritools Undertow distribution.)*

Linking In PLI Routines

Downloading the FSM Example from the Veritools Web Site

1. You can download the FSM examples used in this section, from the Veritools web site:

   www.veritools.com

2. From the main URL, follow the links of *download => Undertow => examples.*

The web site also includes complete documentation for these examples.

Synopsys VCS

You need to link in the Veritools PLI routines, when you compile your model with VCS. Veritools supplies the following files:

- iv.v
- vtplivcs.o
- vt_vcs.tab

The Veritools PLI routines use these files to generate a VCS compiled model.

*Note: **vt_vcs.tab** defines all Veritools user-defined PLI definitions.*

Example

This example model consists of the following files:

- fsm1.v
- fsm2.v
- fsm3.v
- top.v

1. To compile this model, invoke VCS as follows:

   - Normal Mode for Batch:

     vcs +rad +vcsd -Mupdate fsm1.v fsm2.v fsm3.v top.v iv.v
     $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/vt_vcs.tab -lm

   - Normal Mode for Interactive:

     vcs +rad +vcsd -line -Mupdate fsm1.v fsm2.v fsm3.v top.v iv.v
     $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/vt_vcs.tab -lm

     *vcsi -Mupda*te *+rad +vcsd -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab*

     Call up UT with the following syntax:

     ut -iv -vcs <*vcs_simulator_name*> <*simulator_commands*>
     -sigfile <*signal_file_name*> - tracefile <*trace_file_name*>
     ivsimcmp "-f source_code.vc"

     ut -iv -vcs simv -sigfile ut.sigs -ivsimcmp “-f source”

     Change the button file to vcs.btn (File->open->Buttons) and select vcs.btn

     Go to Simulator-> Run

   *Note: You must include the **-line** option in your VCS compilation command, if you want to run Undertow Suite in an **interactive mode**. If you want to run VCS in batch mode, you do not need to put **-line** into the command line.*

   Invoking VCS produces a **simv** executable.

 Copyright Veritools 2004 All Rights Reserved

2. You can then run this executable, as in the following example:

   simv +VTCOMPRESS200

   The simulator runs, and creates a **vt.dump** file.

Sample Script For Running VCS

STEPS for SIMULATING USING VCS:

1)Source environment variables

For example,

envsource has all the environment variables set up. You can change the paths accordingly.

----------*envsource file contents*----------

setenv VCSI_HOME <path where VCSI is installed>

setenv DEFAULT_VCS_HOME <path where VCSI is installed>

setenv VCS_HOME <path where VCSI is installed>

setenv TMPDIR /tmp

setenv VCS_NETHOST vt1

setenv VCS_LTD_LICENSE 1

setenv VCS_CC cc

---

2)Add to source code close to top module

initial

begin

$vtDumpvars();

end

In this example, the above has been inserted in 'top.v' file.

3) To run simulator in batch mode, do as follows:

./run_vcsi_batch

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

run_vcsi_batch : script for running all vcs commands.

---------run_vcsi_batch contents----------

#!/bin/csh -f

*vcsi -Mupda*te *+rad +vcsd -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab*

./simv +VTCOMPRESS250 +VTVECTORVALUES
----

a) -f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v iv.v is available in the example directory.

b) vtplivcs.o and vt_vcs.tab: are available in our PLI directory. You need to link in the Veritools PLI routines when you compile your model with VCS. The Veritools PLI routines use these files to generate a VCS compiled model. vt_vcs.tab defines all Veritools user-defined PLI definitons.

c) -P: used to specify the PLI tab file.

> *NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.*

The simulator runs and creates a vt.dump file.

Compilation:

vcsi <flags> -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/ PLI/vt_vcs.tab

To enhance Simulation speed, users can run VCS wth different options to enhance/hinder performance. Try avoiding the slow option and use the fast options. Here are the options users can use in VCS

Options for for increased performance in VCS compilation:

+rad: Perform aggressive optimization for RTL or (non timing) gate level

+vcsd: Use more efficient Direct Kernel Interface for VirSim dumping

-Mupdate: Use incremental compile to only recompile modules that

functionally changed

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

+rad+1: Perform localized optimizations only

+nospecify: Ignore path delays and timing checks for functional gat simulations

+notimingchecks: Ignore timing checks +timopt: Perform agressive timing optimizations (for SDF gate simulations)

+nocelldefinepli+2: Do not dump or use PLI access from within library cells

+2state: Convert entire design (except certain constructs) to 2 states for more speed/capacity.

Flags that hinder performance:

+cli: Turn on interactive debug for the entire design

+cli+mod=#: Turn on interactive debug just for module mod

+acc: Obsolete flag to allow for global PLI access

-line: Allow line stepping in debugger

-00: Turn off optimizations

-I: Obsolete flag for interactive GUI debug. Instructs VCS to automatically include the +cli (command line interface), -P virsims.tab (default VirSim PLI table) and -lm (math library) options. This option is required to create VCD+ files like the -RI option. The virsims.tab file defines the VCD+ system tasks such as $vcdpluson. Unlike the -RI option, simulation with VirSim does not start after compilation but allows post-processing of the VCD+ file after simulation

-RI: Compile and run interactive debug with VirSim GUI

-PP: Allow VirSim vcd+ binary dumping for post-processing debug. Does the same thing as the -I option but minimizes detail of net data for faster post-processing

+race: Turn on race detection

+prof: Turn on VCS profiling

-gen_c: Force generation of C intermediate code instead of native object code -P pli.tab which contains acc=rw, cbk:*: Globally turn on PLI access +multisource_int_delays: Enables global PLI access visibility +transport_int_delay: Enables global PLI access visibilty-X*: version

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

specific flags to work around specific bugs

Recommended Use(for optimal speed):

vcsi -Mupdate +rad +vcsd -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab

./simv +VTCOMPRESS250 +VTVECTORVALUES

4) To use interactive mode:

./run_vcsi_int

-------run_vcsi_int contents----------

#!/bin/csh -f

vcsi -Mupdate +cli +acc -line -lm -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab

---

The -line option must be given if you want to run Undertow Suite in interactive mode. The -line option is not required for running VCS in batch mode.

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

BATCH MODE:

ut -iv -f <source> -sigfile <dump filename>

NOTE: <source> is the file that contains the source code files for simulation.

For example,

ut -iv -f source -sigfile vt.dump

INTERACTIVE MODE:

ut -iv -vcs <vcs_simulator_executable> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code file names>"

For example,

ut -iv -vcs simv -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

for information about using -tracefile please refer to “Trace Files” on page 410

Cadence Verilog-XL

Note to Cadence LDV User

Undertow 2004.1.7: For Cadence LDV versions 4.1, 5.1, 5.2 and up, you can use the new version of PLI for higher compression.

To replace the old PLI with the new PLI,

> cd $UT_ROOT_DIR/PLI

> mv libpli.so libpli.so ## for future use

>cp libplivpi.so libpli.so

For older versions of Cadence, libpli.so is used by default to generate fast file.

You need to create a new Verilog executable, with the Veritools PLI routines linked in.

Veritools supplies the following files:

- vtpli.o
- vt_veriuser.c

1. Set the **ver_installdir** environment variable, to point to the Verilog installation directory.

   Normally, this is:

   /*cdsdir*/tools/verilog

   where *cdsdir* is the root of your Cadence software distribution tree.

2. Run the **vconfig** utility, supplied with your Verilog distribution.

   The **vconfig** program prompts you for various information, and shows the default values. These defaults are usually the right choices.

   This utility produce a script, which creates a new Verilog executable.

3. The link statement must include the **veriuser.c** user template file.

Example:

- The **vconfig** program displays the following prompt:

  The user template file'veriuser.c' must always be included in the link statement. What is the path name of this file?

  [*/cdsdir/tools/verilog/src/veriuser.c]:*

  To respond to this prompt, enter the complete path name to the **vt_veriuser.c** file.

- The **vconfig** program displays the following prompt:

  Please list any other user files to be linked with this Verilog-XL. Files ending with'.c' will be taken as C source files and compiled before being linked with Verilog-XL. All other files will be added to the final link command as-is. For C++ linking in the Standalone mode, you must manually edit the resultant shell script. List the files one at a time, terminating the list with a single'.'

  To respond to this prompt, enter the complete path name to the **vtpli.o** file.

For a detailed description of the **vconfig** utility, refer to your Verilog documentation. This utility is specific to the version of Verilog that you are currently running. After you run **vconfig**, be sure to execute the script that **vconfig** generates.

Build your Verilog-XL in Static Mode

vconfig displays the following prompt:

Please choose a target.

valid choices: 1) Stand Alone,

2) Backplane,

3) Verilog Export,

4) VHDL Import,

5) Dynamic PLI libraries only,

<Enter 1-5> [1]: 1

Build your Veritools PLI in Dynamic Mode:

vconfig displays the following prompt:

Please choose a target.

valid choices: 1) Stand Alone,

2) Backplane,

3) Verilog Export,

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

4) VHDL Import,

5) Dynamic PLI libraries only,

<Enter 1-5> [1]: 5

Running your Verilog build script, cr_vlog, create a libpli.so file. To run Verilog dynamically, make sure that the libpli.so file is defined in your library search path.

You can setup your environmental variable in 2 ways:

1, setting up environment variable in the start up file(.cshrc)

setenv LD_LIBBRARY_PATH $LD_LIBRARY_PATH:${UT_ROOT_DIR}/PLI

or

2 add the followin to the command line

+loadpli1=$(UT_ROOT_DIR)/PLI/libpli.so:my_bootstrap

Cadence NC Verilog

Steps to build your ncelab and ncsim executable along with Veritools PLI:

1. Add the Veritools **$UT_ROOT_DIR/vt_veriuser.c** and **$UT_ROOT_DIR/vtplinc.o** files to your **Makefile.nc** file.
2. To create your ncelab script:

   >make ncelab

3. To create your ncsim script:

   >make ncsim

Sample Script for running NCSIM:

1)Source environment variables

you can setup your environmental variable in 2 ways:

1, setting up environment variable in the start up file(.cshrc)

setenv LD_LIBBRARY_PATH $LD_LIBRARY_PATH:${UT_ROOT_DIR}/PLI

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

or

2, add the followin to the command line

+loadpli1=$(UT_ROOT_DIR)/PLI/libpli.so:my_bootstrap

For example,

./envsource

envsource has all the environment variables set up. You can change the paths accordingly. Make sure that the $CDS_INST_DIR is defined for your Cadence installation directory.

vt_veriuser.c and vtplinc.o are available in our PLI directory.

----------envsource file contents----------

#setenv CDS_INST_DIR <cadence installation directory>

setenv CDS_INST_DIR /cad_tools/LDV5.1

setenv ittsimUndertowSeDir $CDS_INST_DIR/tools/dfII/local/undertow

setenv LD_LIBRARY_PATH /usr/lib:/usr/openwin/lib:$CDS_INST_DIR/tools/dfII/lib:$CDS_INST_DIR/tools/inca/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/verilog/lib:/usr/dt/lib:/usr/lib/x11:/usr/ucblib:/usr20/dt_cde/lib:/usr/local/lib/gcc-lib:/usr/local/lib:{SILOS}/bin:$UT_ROOT_DIR/PLI

Add to source code close to top module

initial

begin

$vtDumpvars;

end

In this example, the above has been inserted in 'top.v' file.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

If you are simulating your design for the first time, follow these steps:

a) Run "ncprep"

>ncprep -f source

-f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

Note that ncprep will generate the following files and directories.

- *cds.lib*
- *hdl.vars*
- *INCA_LIB*
- *ncvlog.args*
- *ncelab.args*
- *ncsim.args*

b) Run "ncvlog"

>ncvlog -f ncvlog.arg

c) Add the following line into file "ncleab.args"

-ACCESS +RCW

d) Run "ncelab"

>ncelab -f ncelab.args

e) Run "ncsim"

>ncsim -f ncsim.args

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

You can then run the simulator again as follows:

./run_ncsim

run_ncsim : script for running all modelsim commands.

```
---------run_ncsim contents-----------------------------------------
#!/bin/csh -f

# Run the NC-Verilog parser (compile the source)
ncvlog -f ncvlog.args
if ($status != 0) then
  exit
endif

# Run the NC-Verilog elaborator (build the design hierarchy)
ncelab -f ncelab.args
if ($status != 0) then
  exit
endif

# Run the NC-Verilog simulator (simulate the design)
#ncsim -f ncsim.args +VTCOMPRESS250 +VTVECTORVALUES
ncsim -f ncsim.args
-------------------------------------------------------------------------
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size o

Viewing the NC Sim Waveform

Batch Mode

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

source_code_file is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

Interactive Mode

a) Cadence Verilog-XL(non-compiled)

ut -iv -xl verilog -f <file that lists all source code filenames> -sigfile <signal_filename> -tracefile <trace_filename>

For example,

ut -iv -xl verilog -f source -sigfile fsm.sigs -tracefile fsm.trace

b) Cadence NC Verilog-XL *(non-compiled)*

ut -iv -ncxlmode ncxlmode -f <file that lists all source code filenames> -sigfile <signal_filename> -tracefile <trace_filename>

For example,

ut -iv -ncxlmode ncxlmode -f source -sigfile fsm.sigs -tracefile fsm.trace

c, Cadence NC Verilog *(Non-compiled Simulator)*

ut -iv -nc ncverilog "-f <file that lists all source code filenames>" -sigfile <signal_filename> -tracefile <trace_filename>

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

For example,

ut -iv -ncxlmode ncverilog "-f source" -sigfile fsm.sigs -tracefile fsm.trace

d, Cadence NC Sim *(Compiled Simulator)*

ut -iv -nc ncsim "-f ncsim.args" -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code file names>"

For example,

ut -iv -nc ncsim "-f ncsim.args" -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

OR

ut -iv -nc ncsim worklib.top:v -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Note: for information about using -tracefile please refer to “Trace Files” on page 410

Note: For more informations about Compiled Simulator and Non Compiled Simulator. Please refer to “Batch Simulation with Command Line” on page 288

Modeltech Vsim

Set the **PLIOBJS** environment variable, to the same location as **vtplimodtech.so**. **vtplimodtech.so** is included with the distribution in the PLI directory.

Example:

setenv PLIOBJS $UT_ROOT_DIR/PLI/vtplimodtech.so

Script file for running MODELTECH

Sourcing Env. Variable, For example,

./envsource

----------envsource file contents----------

setenv PLIOBJS $UT_ROOT_DIR/PLI/vtpli_modtech.so

#setenv ModelTech <path where ModelTech has been installed>

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

---

vtpli_modtech.so is available in our PLI directory.

Add to source code close to top module

initial

begin

    $vtDumpvars;

end

In this example, the above has been inserted in 'top.v' file.

Run simulator as follows:

./run_modeltech

run_modeltech : script for running all modelsim commands.

---------run_modeltech contents----------

```
#!/bin/csh -f
if (-e work) then
    rm -r -f work
endif
if (! -e work) then
    vlib work   #creates new design library work
endif
```

vlog -f source #compiles the verilog files into the work library

vsim -c -do 'run -all' top vtInteractive +VTCOMPRESS250 +VTVECTORVALUES

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

#vsim -c -do 'run -all' top vtInteractive

---

a) **-f <file> :** used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

b) **-c :** run in command line mode

c) -**do 'run -all'**: to run until there are no more simulation events or until $finish is executed in the Verilog code.

d) **top and vtInteractive:** are the top level modules in this example. The simulator vsim is invoked with the top level modules.

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.

Viewing Model Tech Waveform

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

<source_code_file> is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

INTERACTIVE MODE

ut -iv -modeltech <simulator_executable_name> <top level module names> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f

<file that lists all source code files>"

For example,

ut -iv -modeltech vsim top -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Note: For more information on how to generate a tracefile, please refer to"Trace Files" on page 410 .

Fintronic FinSim

If you compile your model with FinSIm, you need to link in Veritools's PLI routines. Veritools supplies the following files:

- iv.v
- vtplifinsim.o
- vt_finsimuser.c
- vt_ptab

Veritools PLI routines use these files to generate a FinSim-compiled model.

1. Add the following arguments to your current **finvc** command line:

   -a +vtdbg -ptab vt_ptab

   -dsm int -f source

2. Add the following to your **FINUSERPLIOBJ** environment variable:

   - **vtplifinsim.o**
   - **vt_finsimuser.o**

3. For Express-VCT, add *+finvcc* to the **finvc** command line.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

4. Run **vccdump** to obtain the vcc file.

Sample script for running Finsim

1)Make sure you set the FINUSERPLIOBJ environment variable follows:

setenv FINUSERPLIOBJ "<path>/vtplifinsim.o <path>/vt_finsimuser.o"

vt_finsimuser.o and vtplifinsim.o are available in our PLI directory. Make sure the path to the PLI directory is specified.

----------envsource file contents----------

setenv FINTRONIC <path where Finsim is installed>

setenv FIN_OBJECT_PATH $FINTRONIC/obj/gcc

setenv FIN_INCLUDE_PATH $FINTRONIC/include

setenv FIN_LIBRARY_PATH $FINTRONIC/lib/gcc

setenv FINTEMPDIR ./fintmp

setenv FIN_COMPILER gcc

#setenv FINUSERPLIOBJ "<path_to>/vtplifinsim.o <path_to>/vt_finsimuser.o"

#NOTE: The path and the files are within "" separated by a space.

setenv FINUSERPLIOBJ "$UT_ROOT_DIR/PLI/vtplifinsim.o $UT_ROOT_DIR/PLI/vt_finsimuser.o"

#The files are available in our PLI directory.

setenv FIN_LICENSE_PATH $FINTRONIC/env

#setenv _ULKFLAGS "-lnsl -lsocket -lintl"

#Above setenv is for SOLARIS ONLY!

setenv FINSYSPLIOBJ $FIN_OBJECT_PATH/verisim.o

setenv $UT_ROOT_DIR <path_to directory holding Undertow distribution>

---

2)Add to source code close to top module

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
 initial
begin
    $vtDumpvars;
end
```

In this example, the above has been inserted in 'top.v' file.

3)Modify 'top.cf' file, for example:

-a +vtdbg -ptab vt_ptab

-dsm int -f source

-a : suppresses all warning messages

+vtdbg : generates information for Veritool's source level debugger

-ptab <file> : gets information about pli tasks/functions from the specified ascii file instead of having to link user's pli object files to obtain this information

vt_ptab is available in our PLI directory. Make sure the path to this directory is specified.

-dsm <mode> : sets the default simulation mode to compile (-dsm com) or interpret (-dsm int), -dsm com is the default

Precedence order for simulation mode options

The Verilog compiler determines the simulation mode of a module (compiled or interpreted) based on the invocation options `-comm <mod>', `-intm <mod>', `-comf <file>', `-intf <file>', `-comd <dir>', `-intd <dir>' and `-dsm <mode>'. The precedence order of these options are:

1. -comm <mod>, -intm <mod>
2. -comf <file>, -intf <file>
3. -comd <dir>, -intd <dir>
4. -dsm <mode>

For example, if the user types the following:

finvc -intf test.v -comm test

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

and the module `test' is defined in the file `test.v', then all modules in file `test.v' will be interpreted except `test' which will be compiled.

-f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.

4)Run simulator as follows:

./runsim

runsim : script for running all finsim commands.

---------runsim contents----------

```
#!/bin/csh
# Usage: runsim <options>
# This is a sample script which runs the 3 steps required by the
# FinSim products:
#       1) finvc        --- the Verilog compiler and code generator
#       2) finbuild     --- the simulation builder
#       3) TOP.sim      --- the simulator
# Different options for these 3 steps can be specified by using the
# variables FINVCOPT, FINBUILDOPT and FINSIMOPT below. For the compiler
# finvc, options can also be specified in the command file top.cf or
# passed directly in the command line of this script like this:
#               runsim +notimingchecks -dsm int
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
# For a complete description of the available options please see the User
# Guide or run any of these commands with the option -help.

if (!($?FINTRONIC)) then
  echo ""
  echo " The environment variable FINTRONIC is not set. Please source the file"
  echo " env/env.script in the installation directory and rerun this script"
  echo ""
  exit(1)
endif

set path = ($FINTRONIC/bin/$FIN_COMPILER $path)

# set the options for finvc
set FINVCOPT = "-cf top.cf -a -ol 9"
# set the options for finbuild
set FINBUILDOPT = "-verbose"
# set the options for TOP.sim
set FINSIMOPT = "-verbose"

# run the compiler finvc
finvc $FINVCOPT $*
if ($status != 0) then
  echo ""
  echo "***ERROR*** The compiler terminated abnormally. Please check the file finvc.log"
  echo ""
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
  exit(1)
endif

# run the simulation builder finbuild
finbuild $FINBUILDOPT
if ($status != 0) then
  echo ""
  echo "***ERROR*** The simulation builder terminated abnormally. Please check the files finbuild.log and compile.log"
  echo ""
  exit(1)
endif

# run the simulator TOP.sim
./TOP.sim $FINSIMOPT
if ($status != 0) then
  echo ""
  echo "***ERROR*** The simulation terminated abnormally. Please check the file finsim.log"
  echo ""
  exit(1)
endif

echo ""
echo "The simulation has completed successfully"
echo ""
```

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Viewing Finsim Waveform

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

<source_code_file> is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

INTERACTIVE MODE

ut -iv -finsim <simulator_executable> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code files>"

For example,

ut -iv -finsim TOP.sim -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Note: for information about using -tracefile please refer to “Trace Files” on page 410

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

VHDL CADENCE NCSIM

Running Undertow Suite with NCSim

CONTENTS:

I) Generating a VCD file for a pure VHDL design

II) Generating Veritools vt.dump file for a pure Verilog design/pure VHDL design/Mixed VHDL Verilog design using Veritools VPI/VHPI

III) Generating Veritools vt.dump file for a pure Verilog design using Veritools PLI

IV) Summary-using Veritools PLI/VPI/VHPI

1) Linking in PLI/VPI/VHPI for NCSim: Verilog, VHDL and Mixed Verilog/VHDL designs

a) For pure Verilog designs

b) For pure VHDL designs or VHDL/Verilog mixed designs

c) Three ways to run the VPI/VHPI routines with VHDL or mixed VHDL/Verilog designs

V) Loading Design into Undertow Suite

1) For Verilog designs

2) For Verilog/VHDL or pure VHDL designs that are not complied

3) For Verilog/VHDL or pure VHDL designs that are complied with vtcom

4) Support for Cadence cds.lib

VI) Using Veritools Compiler Adapter vtcom for compiling designs without using a makefile:

1) Compiling vendor libraries

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2) Compiling your design using vtcom

3) Loading design into Undertow Suite using these compiled libraries/files

VII) Using Veritools Compiler Adapter vtcom for compiling designs using a makefile:

1) Generating a makefile using Veritools' generic make utility

2) Script for executing makefile with Veritools' vtcom

3) Veritools Compiler Adapter vtcom

4) Loading design into Undertow Suite using these compiled libraries/files

---

I) Generating a VCD file for a pure VHDL design:

Please refer to $UT_ROOT_DIR/FSM_VHDL_EXAMPLE.

1) Make the sure the following environment variables are set:

setenv CDS_INST_DIR <cadence installation directory>

setenv LD_LIBRARY_PATH /usr/lib:/usr/openwin/lib:$CDS_INST_DIR/tools/dfII/lib:$CDS_INST_DIR/tools/inca/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/verilog/lib:/usr/dt/lib:/usr/lib/x11:/usr/local/lib:$UT_ROOT_DIR/PLI

Make sure LD_LIBRARY_PATH has $UT_ROOT_DIR/PLI specified in the path.

For UNDERTOW versions 2004.1.8 and up:

Use the correct PLI according according to the simulator type and version. $UT_ROOT_DIR/PLI has the following .so files:

libpli.so.nc_verilog: For latest CADENCE LDV versions 4.1, 5.1, 5.2 and up

libpli.so.old_nc_verilog: For earlier CADENCE LDV versions

Make sure you do the following:

%cd $UT_ROOT_DIR/PLI/

%cp <the appropriate libpli.so.#####> libpli.so

For UNDERTOW versions 2004.1.7 and below:

For latest CADENCE LDV versions 4.1, 5.1, 5.2 and up:

Make sure you do the following:

%cd $UT_ROOT_DIR/PLI/

%mv libpli.so libpli.so.old

%cp libplivpi.so libpli.so

For earlier CADENCE LDV versions, libpli.so is used by default.

2) Make sure you have the cds.lib and hdl.var files. Copy the cds.lib and hdl.var files in your design directory.

----------cds.lib contents----------

DEFINE worklib ./worklib

SOFTINCLUDE $CDS_INST_DIR/tools/inca/files/cdsvhdl.lib

----------hdl.var contents----------

softinclude $CDS_INST_DIR/tools/inca/files/hdl.var

define NCVHDLOPTS -message -linedebug

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
define NCELABOPTS -message
define NCSIMOPTS  -message
define WORK worklib
-------------------------------------------------
```

3) To create a dump.vcd file using ncsim, run the script as follows:

```
./run_ncsim_vcd
```

dump.vcd is generated.

```
------------run_ncsim_vcd contents--------------------
#!/bin/csh -f
if (-e worklib) then
      rm -r -f worklib
endif
if (! -e worklib) then
      mkdir worklib
endif

ncvhdl -LINEDEBUG -f source
ncelab -MESSAGE -ACCESS +RCW  worklib.top:rtl
ncsim -TCL top -INPUT command_vcd
---------------------------------------------------
```

Here, top is the top level entity and rtl is the top level architecture.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
----------command_vcd contents-------------------
call vcdfile dump.vcd -t ns
call vcdaddscope : -a -r
call vcdon
run 100000ns
call vcdflush
quit
-----------------------------------------------

%./run_ncsim_vcd
ncvhdl: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
fsm1.vhd:
        errors: 0, warnings: 0
fsm2.vhd:
        errors: 0, warnings: 0
fsm3.vhd:
        errors: 0, warnings: 0
top.vhd:
        errors: 0, warnings: 0
WORKLIB.FSM1 (entity):
        streams: 6, words: 24
WORKLIB.FSM1:RTL (architecture):
        streams: 1, words: 237
WORKLIB.FSM2 (entity):
        streams: 6, words: 37
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
WORKLIB.FSM2:RTL (architecture):
        streams: 1, words: 291
WORKLIB.FSM3 (entity):
        streams: 7, words: 56
WORKLIB.FSM3:RTL (architecture):
        streams: 1, words: 239
WORKLIB.TOP (entity):
        streams: 1, words: 9
WORKLIB.TOP:RTL (architecture):
        streams: 1, words: 148
ncelab: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F1) with
design unit (WORKLIB.FSM1:RTL).
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F2) with
design unit (WORKLIB.FSM2:RTL).
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F3) with
design unit (WORKLIB.FSM3:RTL).
        Building instance specific data structures.
        Design hierarchy summary:
                        Instances  Unique
            Components:         5       4
            Default bindings:   3       -
            Processes:         15      15
            Signals:           21      21
        Writing initial simulation snapshot: WORKLIB.TOP:RTL
ncsim: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
```

Loading snapshot worklib.top:rtl ..... Done

ncsim> source /usr32/install_sim/cadence/LDV5.1/tools/inca/files/ncsimrc

ncsim> call vcdfile dump.vcd -t ns

ncsim> call vcdaddscope : -a -r

ncsim> call vcdon

ncsim> run 100000ns

Ran until 100 US + 0

ncsim> call vcdflush

ncsim> quit

II) Generating Veritools vt.dump file for a pure Verilog design/pure VHDL design/Mixed VHDL Verilog design using Veritools VPI/VHPI:

Please refer to $UT_ROOT_DIR/FSM_VHDL_EXAMPLE

1) Make the sure the following environment variables are set:

setenv CDS_INST_DIR <cadence installation directory>

setenv LD_LIBRARY_PATH /usr/lib:/usr/openwin/lib:$CDS_INST_DIR/tools/dfII/lib:$CDS_INST_DIR/tools/inca/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/verilog/lib:/usr/dt/lib:/usr/lib/x11:/usr30/builds/solaris32/2004dbg/2004-vtpli:/usr/local/lib

Make sure LD_LIBRARY_PATH has the path to the directory where vtvpivhpi_nc.so is present.

2) Make sure you have the cds.lib and hdl.var files. Copy the cds.lib and hdl.var files in your design directory. The following is for a pure VHDL design.

----------cds.lib contents----------

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
DEFINE worklib ./worklib

SOFTINCLUDE $CDS_INST_DIR/tools/inca/files/cdsvhdl.lib

-------------------------------------------------
```

----------hdl.var contents----------

```
softinclude $CDS_INST_DIR/tools/inca/files/hdl.var

define NCVHDLOPTS -message -linedebug

define NCELABOPTS -message

define NCSIMOPTS  -message

define WORK worklib
```

---

3) To create a vt.dump file using ncsim and our VPI/VHPI vtvpivhpi_nc.so, run the script as follows:

```
./run_ncsim_vhpi
```

Press Cntrl-C to stop simulation. vt.dump is generated.

----------run_ncsim_vhpi contents----------

```
#!/bin/csh -f

if (-e worklib) then

    rm -r -f worklib

endif

if (! -e worklib) then

    mkdir worklib
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
endif

ncvhdl -LINEDEBUG -f source
ncelab -MESSAGE -ACCESS +RCW worklib.top:rtl
ncsim -loadvhpi vtvpivhpi_nc:vtVhpiDumpvars TOP
-----------------------------------------------------
```

Here, TOP is the top level entity name in the VHDL design.

```
%./run_ncsim_vhpi
ncvhdl: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
fsm1.vhd:
        errors: 0, warnings: 0
fsm2.vhd:
        errors: 0, warnings: 0
fsm3.vhd:
        errors: 0, warnings: 0
top.vhd:
        errors: 0, warnings: 0
WORKLIB.FSM1 (entity):
        streams: 6, words: 24
WORKLIB.FSM1:RTL (architecture):
        streams: 1, words: 237
WORKLIB.FSM2 (entity):
        streams: 6, words: 37
WORKLIB.FSM2:RTL (architecture):
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
        streams: 1, words: 291
WORKLIB.FSM3 (entity):
        streams: 7, words: 56
WORKLIB.FSM3:RTL (architecture):
        streams: 1, words: 239
WORKLIB.TOP (entity):
        streams: 1, words: 9
WORKLIB.TOP:RTL (architecture):
        streams: 1, words: 148
ncelab: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F1) with
design unit (WORKLIB.FSM1:RTL).
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F2) with
design unit (WORKLIB.FSM2:RTL).
ncelab: *W,CUDEFB: default binding occurred for component instance (:TOP(RTL):F3) with
design unit (WORKLIB.FSM3:RTL).
        Building instance specific data structures.
        Design hierarchy summary:
                        Instances  Unique
            Components:         5       4
            Default bindings:   3       -
            Processes:         15      15
            Signals:           21      21
        Writing initial simulation snapshot: WORKLIB.TOP:RTL
ncsim: 05.10-s004: (c) Copyright 1995-2004 Cadence Design Systems, Inc.
Loading snapshot worklib.top:rtl ..... Done
```

vtpli.o and vtplivcs.o, Copyright 2004 Veritools Inc.,

All rights reserved -- Version 2004.1.0

ncsim> source /usr32/install_sim/cadence/LDV5.1/tools/inca/files/ncsimrc

ncsim> run

^CSimulation interrupted at 421883 NS + 2

ncsim> quit

III) Generating Veritools vt.dump file for a pure Verilog design using Veritools PLI

Please refer to the FSM_VERILOG_EXAMPLE directory for the design example.

1)Source environment variables

Make sure that the $CDS_INST_DIR is defined for your Cadence installation directory.

----------envsource file contents----------

#setenv CDS_INST_DIR <cadence installation directory>

setenv CDS_INST_DIR /cad_tools/LDV5.1

setenv ittsimUndertowSeDir $CDS_INST_DIR/tools/dfII/local/undertow

setenv LD_LIBRARY_PATH /usr/lib:/usr/openwin/lib:$CDS_INST_DIR/tools/dfII/lib:$CDS_INST_DIR/tools/inca/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/verilog/lib:/usr/dt/lib:/usr/lib/x11:/usr/ucblib:/usr20/dt_cde/lib:/usr/local/lib/gcc-lib:/usr/local/lib:{SILOS}/bin:$UT_ROOT_DIR/PLI

Make sure LD_LIBRARY_PATH has $UT_ROOT_DIR/PLI/ specified in the path.

NOTE:

For UNDERTOW versions 2004.1.8 and up:

Use the correct PLI according according to the simulator type and version. $UT_ROOT_DIR/PLI has the following .so files:

libpli.so.nc_verilog: For latest CADENCE LDV versions 4.1, 5.1, 5.2 and up

libpli.so.old_nc_verilog: For earlier CADENCE LDV versions

Make sure you do the following:

%cd $UT_ROOT_DIR/PLI/

%cp <the appropriate libpli.so.#####> libpli.so

For UNDERTOW versions 2004.1.7 and below:

For latest CADENCE LDV versions 4.1, 5.1, 5.2 and up:

Make sure you do the following:

%cd $UT_ROOT_DIR/PLI/

%mv libpli.so libpli.so.old

%cp libplivpi.so libpli.so

For earlier CADENCE LDV versions, libpli.so is used by default.

2)Add to source code close to top module

initial

begin

$vtDumpvars();

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

end

In the example in $UT_ROOT_DIR/FSM_VERILOG_EXAMPLE directory, the above has been inserted in 'top.v' file.

3) If you are simulating your design for the first time, follow these steps:

a) Run "ncprep"

>ncprep -f source

-f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v. iv.v is available in the example directory.

Note that ncprep will generate the following files and directories.

cds.lib

hdl.var

INCA_LIB

ncvlog.args

ncelab.args

ncsim.args

b) Run "ncvlog"

>ncvlog -f ncvlog.args

c) Add the following line into file "ncleab.args"

-ACCESS +RCW

d) Run "ncelab"

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

>ncelab -f ncelab.args

e) Run "ncsim"

>ncsim -f ncsim.args

4)You can then run the simulator again as follows:

./run_ncsim

run_ncsim : script for running all ncsim commands.

---------run_ncsim contents---------------------------------------

#!/bin/csh -f

ncprep -f source +overwrite

Run the NC-Verilog parser (compile the source)

ncvlog -f ncvlog.args

if ($status != 0) then

exit

endif

Run the NC-Verilog elaborator (build the design hierarchy)

ncelab -f ncelab.args -ACCESS +RCW

if ($status != 0) then

exit

endif

Run the NC-Verilog simulator (simulate the design)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

ncsim -f ncsim.args

The +overwrite option is used to overwrite the files and directories created earlier by ncprep.

The -ACCESS +RCW turns on the read, connectivity and write access visibility.

BATCH MODE

After you run your simulator and generate the signal files and/or trace files, you can use the Undertow Suite to run batch simulation. Batch simulation debugs your source code without tying up the simulator license.

To open the Undertow Suite in batch mode, the command lines are as follows:

ut -iv -f <source_code_file> -sigfile <signal_file> -tracefile <trace_file>

Here, source_code_file is the file thats lists all the source code files.

You can enter a single file that holds the names of other files (e.g., fsm.vc) or enter a list of design files. For example,

ut -iv -f fsm.vc -sigfile fsm.sigs -tracefile fsm.trace

ut -iv top.v fsm1.v fsm2.v fsm3.v -sigfile fsm.sigs

ut -iv top.vhd fsm1.vhd fsm2.vhd fsm3.vhd -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

INTERACTIVE MODE FOR NCSIM

In the interactive mode, you can run ncsim using the Undertow suite.

To open the Undertow Suite in interactive mode, the command line is as follows:

ut -iv -nc ncsim "-f ncsim.args" -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code file names>"

For example,

ut -iv -nc ncsim "-f ncsim.args" -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

OR

ut -iv -nc ncsim worklib.top:v -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

IV) Summary-using Veritools PLI/VPI/VHPI:

Undertow Suite 2004 now provides full support for debugging VHDL and Verilog designs with Cadence NCSIM 5.1 and later.

Users can use the Veritools PLI routines for generating waveform files from Verilog designs and can use the VPI/VHPI routines for generating waveform files from VHDL and Mixed Verilog/ VHDL designs using NCsim. The VPI/VHPI routines can also be used to generate waveform files using NCsim.

1) Linking in PLI/VPI/VHPI for NCSim: Verilog, VHDL and Mixed Verilog/VHDL designs

a) For pure Verilog designs:

Please refer to the following README's according to the simulator vendor:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

$UT_ROOT_DIR/FSM_VERILOG_EXAMPLE/FSM_CADENCE_README

Use $vtDumpvars() ONLY for pure Verilog designs using the PLI routine.
Link the appropriate PLI for the appropriate simulator.

You can also refer to:
http://www.veritools.com/literature/FSM_README.html

b) For pure VHDL designs or VHDL/Verilog mixed designs:

You can refer to the README's in FSM_VHDL_EXAMPLE directory for further directions and an example (README_run_ncsim_vhpi).
Use ONLY the VPI/VHPI routines ($UT_ROOT_DIR/PLI/vtvpivhpi_nc.so) for pure VHDL or mixed Verilog/VHDL designs.

For Cadence, setenv LD_LIBRARY_PATH $UT_ROOT_DIR/PLI:${LD_LIBRARY_PATH}

c) Three ways to run the VPI/VHPI routines with VHDL or mixed VHDL/Verilog designs:

For Cadence, setenv LD_LIBRARY_PATH $UT_ROOT_DIR/PLI:${LD_LIBRARY_PATH}

#1. Command line:

Cadence vhpi command line:
($UT_ROOT_DIR/FSM_VHDL_EXAMPLE/README_run_ncsim_vhpi)
For Cadence NCSim -loadvhpi vtvpivhpi_nc:vtVhpiDumpvars is added to

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

command line as follows:

%ncsim -loadvhpi vtvpivhpi_nc:vtVhpiDumpvars TOP

This will dump out everything into the dumpfile. This will work with verilog, vhdl, and mixed Verilog/VHDL designs. No arguments can be passed to the routine to limit the scope of the dump.

For an example, please refer to FSM_VHDL_EXAMPLE:
README_run_ncsim_vhpi

#2. $vtVpiDumpvars()

Use this command for NCSim

Add the above command to your source code (top level verilog source file) in an initial block.
This command takes the same arguments as the PLI version $vtDumpvars.
This will dump out both vhdl and verilog. This can ONLY be put in verilog source code, so it can obviously be used only with verilog and mixed verilog vhdl designs. (This is just a matter of getting the routine to be invoked. Once invoked, it can handle both verilog and vhdl). Also, for pure Verilog designs, you can use $vtDumpvars() with PLI routine libpli.so instead.

#3. call vtVpiDumpvars

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Use this command for NCSim.

This can be used at the interactive command prompt for ncsim (or in a

command script file.) It will take arguments just like $vtDumpvars, and will work for Verilog, VHDL , and mixed Verilog/VHDL designs.

V) Loading Design into Undertow Suite:

Once the files are compiled and a waveform file has been written out by

the simulator, use the following commands to load in the source code files and

waveform file into Undertow Suite.

1) For Verilog designs:

%ut -iv -f source -sigfile file_name.dump

2) For Verilog/VHDL or pure VHDL designs that are not complied:

%ut -iv -f source -sigfile file_name.dump

3) For Verilog/VHDL or pure VHDL designs that are complied with vtcom:

%ut -iv -use_ut_libs <path to ut_libs> -sigfile file_name.dump

4) Support for Cadence cds.lib

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Undertow supports the cds.lib declarations used by NCSIM. cds.lib contains all user design library declarations. For example, the cds.lib file can be defined as follows with the DEFINE keyword followed by the library/directory name and then the actual path to that library. The cds.lib file can also have INCLUDE and SOFTINCLUDE statements defined with the keyword INCLUDE or SOFTINCLUDE followed by the actual path to any other cds.lib file that has to be included.

Sample cds.lib file:

---

define worklib /usr/demoacc/FSM_VHDL/INCA_libs/worklib

DEFINE std /usr32/install_sim/cadence/LDV5.1/tools/inca/files/STD

DEFINE ieee /usr32/install_sim/cadence/LDV5.1/tools/inca/files/IEEE

DEFINE vital_memory /usr32/install_sim/cadence/LDV5.1/tools/inca/files/VITAL_MEMORY

DEFINE ncutils /usr32/install_sim/cadence/LDV5.1/tools/inca/files/NCUTILS

DEFINE cds_assertions /usr32/install_sim/cadence/LDV5.1/tools/inca/files/CDS_ASSERTIONS

include /usr32/install_sim/cadence/LDV5.1/tools/inca/files/cdsvhdl.lib

---

The -CDSLIB command line option can be used with Undertow as follows:

%ut -iv -f <source_code_file_list> -sigfile <signal_file> -CDSLIB <path_to_cds.lib_file>

For example,

%ut -iv -f source -sigfile vt.dump -CDSLIB ./cds.lib

VI) Using Veritools Compiler Adapter vtcom for compiling designs without using a makefile:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1) Compiling vendor libraries:

Xilinx supplies the VHDL source files for the XilinxCoreLib library. However, some of these files depend on others. So, the order of the "vtcom" commands are important. In addition, some of the files use the IEEE "math_real" package, which is not a certified standard yet and does not ship with the standard Veritools precompiled libraries.

So, you need to compile this "math_real.vhd" prior to compiling simprim and the xilinxcorelib libraries. We have added the vtcom compile command for the same prior to your command lines which is required for this to work. Also, compiling these files in a particular order is required.

Please make sure your packages and files are compiled in the correct order. If a package is used in an entity or in an architecture or in another package, that package has to be compiled in the appropriate library first.

Likewise, an entity has to be compiled before the corresponding architecture. In many cases, the entity and the architecture are in the same file, with the entity first, so this requirement is met most of the time.

Architectures can be analyzed in any order.

Thus in general, this means that you have to compile the packages first (using vtcom) and if a package is used in another package, that package has to be compiled first, the entities next and the architectures last.

2) Compiling your design using vtcom:

-- MTI Compiler

```
vtcom -mti_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...
```

-- CDN Compiler

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

vtcom -cdn_vhdl -WORK <lib_name> <file1.vhd> <file2.vhd> ...

-- SNPS Compiler

vtcom -snps_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

NOTE: By default, vtcom writes compiled files to ./ut_libs/work. i.e. if vtcom is used as follows:

vtcom -cdn_vhdl <file1.vhd> <file2.vhd> ...

If you specify the -WORK <lib_name> option, the compiled files are written to ./ut_libs/ <lib_name> instead of ./ut_libs/work.

Please use the -WORK option in upper case ONLY for CADENCE:

For example:

vtcom -cdn_vhdl -WORK ieee math_real.vhd

vtcom -cdn_vhdl -WORK simprim simprim_Vcomponents.vhd simprim_Vpackage.vhd simprim_VITAL.vhd simprim_SMODEL.vhd

vtcom -cdn_vhdl -WORK xilinxcorelib blk_mem.vhd

vtcom -cdn_vhdl -WORK dut_lib my_design_ent.vhd my_design_arch.vhd my_support_file.vhd

vtcom -cdn_vhdl -WORK dut_tb_lib dut_tb_ent.vhd dut_tb_arch.vhd

This will allow you to compile the simprim and xilinxcorelib libraries provided the .vhd files in these libraries are compiled in the correct order. The correct order is defined in the Xilinx's website:

http://toolbox.xilinx.com/docsan/xilinx4/data/docs/sim/simu5.html#998650

3) Loading Undertow using these compiled libraries/files:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
%ut -iv -use_ut_libs ./ut_libs -sigfile vt.dump -CDSLIB ./cds.lib
```

where, cds.lib file for example looks as follows:

```
DEFINE worklib ./worklib
DEFINE <lib_name> <path_to_library>
DEFINE simprim <path to simprim>
DEFINE xilinxcorelib <path to xilinxcorelib>
SOFTINCLIDE <path>/cdsvhdl.lib
```

The Cadence nclaunch GUI allows you to create a cds.lib file automatically without having to type in the contents manually.

VII) Using Veritools Compiler Adapter vtcom for compiling designs using a makefile:

1) Generating a makefile using Veritools' generic make utility:

Create a Makefile for your design by using a Makefile generator. Say, for example, you can use Veritools' generic make for the same or you can create one manually for your design using some template.

The script "my_make.pl" is the same as "my_make_ncvhdl.pl" (Creates a makefile for Cadence ncvhdl by default).

a) Make sure you specify the correct path for perl in the 1st line of the perl script file my_make.pl:

The path used is as follows:

```
#!/usr/bin/perl -w
```

b) Make sure your source files have .vhd extension (used in script).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

c) %./run_generic_make

Displays the following and creates makefile_generic.

#

This GNU makefile was generated by ./my_make.pl from makeshadow

#

Can't open makeshadow: No such file or directory at ./my_make.pl line 37.

Ignore the above warning.

--------run_generic_make contents----------

./my_make.pl input > makefile_generic_ncvhdl #to create the makefile

./my_make.pl makeshadow #to create the shadow directory

--------input contents----------

PRIMARY_TARGET top.vhd

top.vhd DEPENDS_ON fsm1.vhd fsm2.vhd fsm3.vhd

Here "input" is the file that specifies the source code files and their dependencies. Please mention the keyword PRIMARY TARGET and then the top level .vhd file(s). You can then express the dependencies using DEPENDS_ON keyword as shown above where top.vhd depends on fsm1.vhd, fsm2.vhd and fsm3.vhd.

d) The my_make.pl script generates a Makefile for vhdl sources. After creating

the Makefile, you need to do the following:

./my_make.pl makeshadow //Read below to see how this works

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

touch *.vhd

e) Motivation

Makefiles are easy to make for C programs because the .o files are deposited into a standard place in a standard way. Additionally, the .o file name depends only on the C file name, not on the file contents.

VHDL object files, on the other hand, are very analyzer-specific, and writing analyzer-specific Makefiles by hand is annoying.

This tool takes a Makefile-like input file and generates an analyzer-specific Makefile automatically. It currently generates a Makefile for use with Cadence ncvhdl. To use with Veritools vtcom, please refer to README_run_vhdl_vtcom_ncvhdl. You can trivially change this.

f) How it works

The input file to make is semantically equivalent to a Makefile, with targets and dependencies. Deliberately the syntax is made different (with the PRIMARY_TARGET and DEPENDS_ON keywords) to make the input files different from actual Makefiles.

A shadow directory (.makeshadow) is created which contains a zero-length file that is updated each time a successful vhdl analysis is

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

done on a vhdl file.

The shadow directory entries are the make targets, and each time an actual VHDL file is touched, the corresponding shadow directory entry becomes out of date. After a successful vhdl analysis invocation, one of the generated rules retouches the shadow directory entry.

The generated Makefile uses the GNU Make $(\@F) variable. Vendor-specific makes may not support this variable.

2) Script for executing makefile with Veritools' vtcom

Make sure you have the "vhdl_packages" available. They should also be available as a part of your installation directory.

Run the script as follows:

./run_vhdl_vtcom_ncvhdl

ut_libs directory is created.

-------run_vhdl_vtcom_ncvhdl contents----------

/usr/local/bin/make -f makefile_generic_ncvhdl VCOM="vtcom -cdn_vhdl" VLOG="vtcom -cdn_verilog" LIB_WORK="./ut_libs/work"

---

3) VERITOOLS COMPILER ADAPTER (vtcom)

For help on using "vtcom", just type vtcom at your command prompt and press enter.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

%vtcom

- Users can run their makefile using the special options listed below including the specification to use vtcom instead of their own compiler.

- VTCOM creates a ut_libs directory that contains Verific translated tcl commands to load the design and search directives for the precompiled vhdl libraries (including the design).

Supports:

-- MTI Compiler (invoke with -mti_verilog or -mti_vhdl)

MTI Makefile should be invoked like this:

make -f makefile VCOM="vtcom -mti_vhdl" VLOG="vtcom -mti_verilog"

-- CDN Compiler (invoke with -cdn_verilog or -cdn_vhdl)

CDN Makefile should be invoked like this:

make -f makefile ncvhdl="vtcom -cdn_vhdl" ncvlog="vtcom -cdn_verilog"

-- SNPS Compiler (invoke with -snps_verilog or -snps_vhdl)

SNPS Makefile should be invoked like this:

make -f makefile vhdlan="vtcom -snps_vhdl" vcs="vtcom -snps_verilog" vcsi="vtcom -snps_verilog"

Output:

- Creates a ./ut_libs directory. If UT_WORK_DIR is defined, then creates a $UT_WORK_DIR/ut_libs directory

- Creates a ./ut_libs/commands.tcl file which contains Undertow ordered-flat compiler directives (Verilog-VHDL)

- Creates a ./ut_libs/libs file which contains Untertow ordered list of libraries for compilation (Verilog-VHDL)

- Creates a ./ut_libs/{libs}/commands.tcl file which contains Untertow ordered per-library compiler directives (Verilog-VHDL)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Creates precompiled standard package libraries ./ut_libs/{ieee, std, synopsys, arithmetic, qsim_logic}/*.vdb if VHDL is used

- Creates precompiled design libraries under ./ut_libs/{libs}/*.vdb if VHDL is used

Settings:

- Environment variable UT_ROOT_DIR has to be defined.

- Environment variable UT_WORK_DIR may be defined to compile in a different directory from the current directory.

- Environment variable UT_STANDARD_VHDL may be defined to fetch user's STANDARD VHDL packages

Undertow provides "vtcom" to analyze a design in "work" library and then elaborate the design just analyzed. "vtcom" mimics analysis and elaboration behavior. The source files will be analyzed and then written to the "work" library in the ut_libs directory. Then, elaboration will be done. Each analyze step can use a different library but will be put in the same "work" directory. The makefile will be made so that users can do incremental analyze steps just on the files that changed and not the whole design.

4) Loading design into Undertow Suite using these compiled libraries:

To use with vtcom, you must start the ut binary in the same directory where you ran vtcom.

%ut -iv -use_ut_libs <path to ut_libs>

For example,

%ut -iv -use_ut_libs ./ut_libs

Undertow will then:

- source the "search_lib.tcl" (for VHDL)

- read the "libs" file for the order of library compilation,

- source all the the "ut_libs/{libs}/commands.tcl" file given their existence (Verilog libraries)

- eventually source the top commands.tcl in case no Verilog files

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- elaborate the top module

NOTE: The <ut_libs path defined for vtcom>, is the path to the ut_libs directory that had been used by vtcom to write out the compiled files and commands. This directory will be used by the "ut -iv" command to restore the compiled files into Undertow Suite. This path is

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

VHDL-MODELTECH

Running Undertow Suite with ModelTech

CONTENTS:

I) Generating a .wlf file for a pure VHDL design/Mixed VHDL Verilog design

II) Generating Veritools vt.dump file for a pure Verilog design using Veritools PLI

III) Summary-using Veritools PLI

1) Linking in PLI for ModelSim: Verilog, VHDL and Mixed Verilog/VHDL designs

a) For pure Verilog designs

b) For pure VHDL designs or VHDL/Verilog mixed designs

IV) Loading design into Undertow Suite

1) For Verilog designs

2) For Verilog/VHDL or pure VHDL designs that are not complied

3) For Verilog/VHDL or pure VHDL designs that are complied with vtcom

V) Using Veritools Compiler Adapter vtcom for compiling designs without using a makefile:

1) Compiling vendor libraries

2) Compiling your design using vtcom

3) Loading design into Undertow Suite using these compiled libraries

VI) Using Veritools Compiler Adapter vtcom for compiling designs using a makefile:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

1) Generating a makefile using ModelSim's vmake utility

2) Script for executing makefile with Veritools' vtcom

3) Veritools Compiler Adapter vtcom

4) Loading design into Undertow Suite using these compiled libraries

---

I) Generating a .wlf file for a pure VHDL design/Mixed VHDL Verilog design:

Please refer to $UT_ROOT_DIR/FSM_VHDL_EXAMPLE.

1) Make sure the following environment variables are set:

setenv PLIOBJS $UT_ROOT_DIR/PLI/vtpli_modtech.so

setenv ModelTech <path where ModelTech has been installed>

vtpli_modtech.so is available in our PLI directory.

2) To create a vsim.wlf file, run the script as follows:

./run_mti_wlf

At the VSIM prompt, do the foll:

VSIM 1> add wave -r /*

VSIM 2> run 50000ns

VSIM 3> quit

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

vsim.wlf file is generated.

```
-------run_mti_wlf contents----------------------
#!/bin/csh -f
if (-e work) then
      rm -r -f work
endif
if (! -e work) then
      vlib work
endif

vcom -f source
vsim -c top
-----------------------------------------

%./run_mti_wlf
Model Technology ModelSim SE vcom 5.8c Compiler 2004.03 Mar 25 2004
-- Loading package standard
-- Loading package std_logic_1164
-- Compiling entity fsm1
-- Compiling architecture rtl of fsm1
-- Compiling entity fsm2
-- Compiling architecture rtl of fsm2
-- Loading package numeric_std
-- Compiling entity fsm3
-- Compiling architecture rtl of fsm3
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
-- Compiling entity top
-- Compiling architecture rtl of top
-- Loading entity fsm1
-- Loading entity fsm2
-- Loading entity fsm3
Reading /usr32/install_sim/modeltech/5.8c/modeltech/tcl/vsim/pref.tcl

# 5.8c

# vsim -c top
# Loading /usr31/distribution/solaris32/ut2004.2.0/PLI/vtpli_modtech.so
# //  ModelSim SE 5.8c Mar 01 2004 SunOS 5.7
# //
# //  Copyright Model Technology, a Mentor Graphics Corporation company, 2004
# //                  All Rights Reserved.
# //              UNPUBLISHED, LICENSED SOFTWARE.
# //       CONFIDENTIAL AND PROPRIETARY INFORMATION WHICH IS THE
# //       PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS.
# //
# Loading /usr32/install_sim/modeltech/5.8c/modeltech/sunos5/../std.standard
# Loading /usr32/install_sim/modeltech/5.8c/modeltech/sunos5/../ieee.std_logic_1164(body)
# Loading /usr32/install_sim/modeltech/5.8c/modeltech/sunos5/../ieee.numeric_std(body)
# Loading work.top(rtl)
# Loading work.fsm1(rtl)
# Loading work.fsm2(rtl)
# Loading work.fsm3(rtl)
```

VSIM 1> add wave -r /*

VSIM 2> run 50000ns

VSIM 3> quit

Veritools currently supports the VPI/VHPI standard that has been supported by Synopsys and Cadence for Verilog, VHDL and Mixed Verilog VHDL designs. We will provide the VPI/VHPI routines once ModelSim also supports the VPI/VHPI. We do not support the FLI.

II) Generating Veritools vt.dump file for a pure Verilog design using Veritools PLI

1)Source environment variables

Make the sure the following environment variables are set:

setenv PLIOBJS $UT_ROOT_DIR/PLI/vtpli_modtech.so

#setenv ModelTech <path where ModelTech has been installed>

setenv ModelTech /cad_tools/modeltech/5.8b/modeltech/bin/

vtpli_modtech.so is available in our PLI directory.

2) Add to source code close to top module

initial

begin

$vtDumpvars();

end

In this example, the above has been inserted in 'top.v' file.

3) Run simulator as follows:

./run_modeltech

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

run_modeltech : script for running all modeltech commands.

---------run_modeltech contents----------

```
#!/bin/csh -f

if (-e work) then

rm -r -f work

endif


if (! -e work) then

vlib work #creates new design library work

endif


vlog -f source #compiles the verilog files into the work library


vsim -c -do 'run -all' top vtInteractive +VTCOMPRESS250 +VTVECTORVALUES

#vsim -c -do 'run -all' top vtInteractive
```

---

a) -f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v. iv.v is available in the example directory.

b) -c : run in command line mode.

c) -do 'run -all': to run until there are no more simulation events or until $finish is executed in the Verilog code.

d) top and vtInteractive are the top level modules in this example. The simulator vsim is invoked with the top level modules.

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file. Optimization is more in the case of designs with large number of vectors.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

BATCH MODE

After you run your simulator and generate the signal files and/or trace files, you can use the Undertow Suite to run batch simulation. Batch simulation debugs your source code without tying up the simulator license.

To open the Undertow Suite in batch mode, the command lines are as follows:

ut -iv -f <source_code_file> -sigfile <signal_file> -tracefile <trace_file>

Here, source_code_file is the file thats lists all the source code files.

You can enter a single file that holds the names of other files (e.g., fsm.vc) or enter a list of design files. For example,

ut -iv -f fsm.vc -sigfile fsm.sigs -tracefile fsm.trace

ut -iv top.v fsm1.v fsm2.v fsm3.v -sigfile fsm.sigs

ut -iv top.vhd fsm1.vhd fsm2.vhd fsm3.vhd -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

INTERACTIVE MODE FOR VSIM (MODELTECH)

In the interactive mode, you can run vsim using the Undertow suite.

To open the Undertow Suite in interactive mode, the command line is as follows:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

ut -iv -modeltech <simulator_executable_name> <top level module names> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code files>"

For example,

ut -iv -modeltech vsim top -sigfile fsm.sigs -ivsimcmp "-f source"

III) Summary-using Veritools PLI:

Undertow Suite 2004 now provides full support for debugging VHDL and Verilog designs with ModelSim 5.8c and later.

Users can use the Veritools PLI routines for generating waveform files for Verilog designs.

1) Linking in PLI for ModelSim: Verilog, VHDL and Mixed Verilog/VHDL designs:

a) For pure Verilog designs:

Please refer to the following README's according to the simulator vendor:

$UT_ROOT_DIR/FSM_VERILOG_EXAMPLE/FSM_MODELSIM_README

Use $vtDumpvars() ONLY for pure Verilog designs using the PLI routine.

Link the appropriate PLI for the appropriate simulator.

You can also refer to:

http://www.veritools.com/literature/FSM_README.html

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

b) For pure VHDL designs or VHDL/Verilog mixed designs:

You can refer to the README's in FSM_VHDL_EXAMPLE directory for further

directions and an example (README_run_mti_wlf).

This will work with verilog, vhdl, and mixed Verilog/VHDL designs and a .wlf file can be generated.

For an example, please refer to FSM_VHDL_EXAMPLE:

README_run_mti_wlf

Veritools currently supports the VPI/VHPI standard that has been supported by Synopsys and Cadence for Verilog, VHDL and Mixed Verilog VHDL designs. We will provide the VPI/VHPI routines once ModelSim also supports the VPI/VHPI. We do not support the FLI.

IV) Loading design into Undertow Suite:

Once the files are compiled and a waveform file has been written out by

the simulator, use the following commands to load in the source code files and

waveform file into Undertow Suite.

1) For Verilog designs:

```
%ut -iv -f source -sigfile file_name.dump
```

2) For Verilog/VHDL or pure VHDL designs that are not complied:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
%ut -iv -f source -sigfile file_name.dump
```

3) For Verilog/VHDL or pure VHDL designs that are complied with vtcom:

```
%ut -iv -use_ut_libs  <path to ut_libs> -sigfile file_name.dump
```

V) Using Veritools Compiler Adapter vtcom for compiling designs without using a makefile:

1) Compiling vendor libraries:

Xilinx supplies the VHDL source files for the XilinxCoreLib library. However, some of these files depend on others. So, the order of the "vtcom" commands are important. In addition, some of the files use the IEEE "math_real" package, which is not a certified standard yet and does not ship with the standard Veritools precompiled libraries.

So, you need to compile this "math_real.vhd" prior to compiling simprim and the xilinxcorelib libraries. We have added the vtcom compile command for the same prior to your command lines which is required for this to work. Also, compiling these files in a particular order is required.

Please make sure your packages and files are compiled in the correct order. If a package is used in an entity or in an architecture or in another package, that package has to be compiled in the appropriate library first.

Likewise, an entity has to be compiled before the corresponding architecture. In many cases, the entity and the architecture are in the same file, with the entity first, so this requirement is met most of the time.

Architectures can be analyzed in any order.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Thus in general, this means that you have to compile the packages first (using vtcom) and if a package is used in another package, that package has to be compiled first, the entities next and the architectures last.

2) Compiling your design using vtcom:

-- MTI Compiler

vtcom -mti_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

NOTE: By default, vtcom writes compiled files to ./ut_libs/work. i.e. if vtcom is used as follows:

vtcom -cdn_vhdl <file1.vhd> <file2.vhd> ...

If you specify the -WORK <lib_name> option, the compiled files are written to ./ut_libs/ <lib_name> instead of ./ut_libs/work.

For example:

vtcom -cdn_vhdl -work ieee math_real.vhd

vtcom -cdn_vhdl -work simprim simprim_Vcomponents.vhd simprim_Vpackage.vhd simprim_VITAL.vhd simprim_SMODEL.vhd

vtcom -cdn_vhdl -work xilinxcorelib blk_mem.vhd

vtcom -cdn_vhdl -work dut_lib my_design_ent.vhd my_design_arch.vhd my_support_file.vhd

vtcom -cdn_vhdl -work dut_tb_lib dut_tb_ent.vhd dut_tb_arch.vhd

This will allow you to compile the simprim and xilinxcorelib libraries provided the .vhd files in these libraries are compiled in the correct order. The correct order is defined in the Xilinx's website:

http://toolbox.xilinx.com/docsan/xilinx4/data/docs/sim/simu5.html#998650

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3) Loading design into Undertow Suite using these compiled libraries:

```
%ut -iv -use_ut_libs ./ut_libs -sigfile vt.dump
```

VI) Using Veritools Compiler Adapter vtcom for compiling designs using a makefile:

1) Generating a makefile using ModelSim's vmake:

Create a Makefile for your design by using your Makefile generator. Say, for example, you can use ModelSim's vmake or you can create one manually for your design using some template.

Before you use vmake, make sure you have compiled your design into the default "work" directory as follows:

vlib work

vcom -f source

Please refer to README_run_mti for more information on the above.

vmake > makefile

The above command redirects the output to "makefile".

ModelSim vmake:

A "makefile" can be generated for your working library with the command: vmake. Executing this makefile causes all out-of-date VHDL files in your library to be recompiled. The vmake utility allows you to use the UNIX MAKE program to maintain libraries. The vmake utility prints a makefile for a library to the standard output.

vmake [-help] [<library_name>] [> <makefile_name>]

For example,

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

i) vmake -help

displays the usage information of vmake.

ii) vmake >my_makefile

produces a make file "my_makefile" for the default library "work"

ii) vmake my_lib >my_makefile

produces a make file "my_makefile" for the "my_lib" library.

2) Script for executing makefile with Veritools' vtcom:

Make sure you have the "vhdl_packages" available. They should also be available as a part of your installation directory.

Run the script as follows:

./run_vhdl_vtcom_mti

ut_libs directory is created.

-------run_vhdl_vtcom_mti contents----------

/usr/local/bin/make -f makefile VCOM="vtcom -mti_vhdl" VLOG="vtcom -mti_verilog" LIB_WORK="./ut_libs/work"

---

3) VERITOOLS COMPILER ADAPTER (vtcom)

For help on using "vtcom", just type vtcom at your command prompt and press enter.

%vtcom

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Users can run their makefile using the special options listed below including the specification to use vtcom instead of their own compiler.

- VTCOM creates a ut_libs directory that contains Verific translated tcl commands to load the design and search directives for the precompiled vhdl libraries (including the design).

Supports:

-- MTI Compiler (invoke with -mti_verilog or -mti_vhdl)

MTI Makefile should be invoked like this:

make -f makefile VCOM="vtcom -mti_vhdl" VLOG="vtcom -mti_verilog"

-- CDN Compiler (invoke with -cdn_verilog or -cdn_vhdl)

CDN Makefile should be invoked like this:

make -f makefile ncvhdl="vtcom -cdn_vhdl" ncvlog="vtcom -cdn_verilog"

-- SNPS Compiler (invoke with -snps_verilog or -snps_vhdl)

SNPS Makefile should be invoked like this:

make -f makefile vhdlan="vtcom -snps_vhdl" vcs="vtcom -snps_verilog" vcsi="vtcom -snps_verilog"

Output:

- Creates a ./ut_libs directory. If UT_WORK_DIR is defined, then creates a $UT_WORK_DIR/ut_libs directory

- Creates a ./ut_libs/commands.tcl file which contains Undertow ordered-flat compiler directives (Verilog-VHDL)

- Creates a ./ut_libs/libs file which contains Undertow ordered list of libraries for compilation (Verilog-VHDL)

- Creates a ./ut_libs/{libs}/commands.tcl file which contains Undertow ordered per-library compiler directives (Verilog-VHDL)

- Creates precompiled standard package libraries ./ut_libs/{ieee, std, synopsys, arithmetic, qsim_logic}/*.vdb if VHDL is used

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- Creates precompiled design libraries under ./ut_libs/{libs}/*.vdb if VHDL is used

Settings:

- Environment variable UT_ROOT_DIR has to be defined.

- Environment variable UT_WORK_DIR may be defined to compile in a different directory from the current directory.

- Environment variable UT_STANDARD_VHDL may be defined to fetch user's STANDARD VHDL packages

Undertow provides "vtcom" to analyze a design in "work" library and then elaborate the design just analyzed. "vtcom" mimics analysis and elaboration behavior. The source files will be analyzed and then written to the "work" library in the ut_libs directory. Then, elaboration will be done. Each analyze step can use a different library but will be put in the same "work" directory. The make file will be made so that users can do incremental analyze steps just on the files that changed and not the whole design.

4) Loading design into Undertow Suite using these compiled libraries:

To use with vtcom, you must start the ut binary in the same directory where you ran vtcom.

%ut -iv -use_ut_libs <path to ut_libs>

For example,

%ut -iv -use_ut_libs ./ut_libs

Undertow will then:

- source the "search_lib.tcl" (for VHDL)

- read the "libs" file for the order of library compilation,

- source all the the "ut_libs/{libs}/commands.tcl" file given their existence (Verilog libraries)

- eventually source the top commands.tcl in case no Verilog files

- elaborate the top module

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

COMPILING VHDL LIBRARIES

Compiling Xilinx libraries:

Xilinx supplies the VHDL source files for the XilinxCoreLib library. However, some of these files depend on others. So, the order of the "vtcom" commands are important. In addition, some of the files use the IEEE "math_real" package, which is not a certified standard yet and does not ship with the standard Veritools precompiled libraries.

So, you need to compile this "math_real.vhd" prior to compiling simprim and the xilinxcorelib libraries. We have added the vtcom compile command for the same prior to your command lines which is required for this to work. Also, compiling these files in a particular order is required.

Please make sure your packages and files are compiled in the correct order. If a package is used in an entity or in an architecture or in another package, that package has to be compiled in the appropriate library first.

Likewise, an entity has to be compiled before the corresponding architecture. In many cases, the entity and the architecture are in the same file, with the entity first, so this requirement is met most of the time.

Architectures can be analyzed in any order.

Thus in general, this means that you have to compile the packages first (using vtcom) and if a package is used in another package, that package has to be compiled first, the entities next and the architectures last.

1) Compile your design as follows for use with the Undertow Suite:

-- MTI Compiler

vtcom -mti_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

-- CDN Compiler

vtcom -cdn_vhdl -WORK <lib_name> <file1.vhd> <file2.vhd> ...

-- SNPS Compiler

vtcom -snps_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

IMPORTANT:

NOTE: By default, vtcom writes compiled files to ./ut_libs/work. i.e. if vtcom is used as follows:

vtcom -cdn_vhdl <file1.vhd> <file2.vhd> ...

If you specify the -WORK <lib_name> option, the compiled files are written to ./ut_libs/ <lib_name> instead of ./ut_libs/work.

Please use the -WORK option in upper case ONLY for CADENCE:

For example:

vtcom -cdn_vhdl -WORK ieee math_real.vhd

vtcom -cdn_vhdl -WORK simprim simprim_Vcomponents.vhd simprim_Vpackage.vhd simprim_VITAL.vhd simprim_SMODEL.vhd

vtcom -cdn_vhdl -WORK xilinxcorelib blk_mem.vhd

vtcom -cdn_vhdl -WORK dut_lib my_design_ent.vhd my_design_arch.vhd my_support_file.vhd

vtcom -cdn_vhdl -WORK dut_tb_lib dut_tb_ent.vhd dut_tb_arch.vhd

This will allow you to compile the simprim and xilinxcorelib libraries provided the .vhd files in these libraries are compiled in the correct order. The correct order is defined in the Xilinx's website:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

http://toolbox.xilinx.com/docsan/xilinx4/data/docs/sim/simu5.html#998650

To bring up Undertow using these compiled libraries/files, use the following:

%ut -iv -use_ut_libs ./ut_libs -sigfile vt.dump -CDSLIB ./cds.lib

where, cds.lib file for example looks as follows:

DEFINE worklib ./worklib

DEFINE <lib_name> <path_to_library>

DEFINE simprim <path to simprim>

DEFINE xilinxcorelib <path to xilinxcorelib>

SOFTINCLIDE <path>/cdsvhdl.lib

The Cadence nclaunch GUI allows you to create a cds.lib file automatically without having to type in the contents manually.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CADENCE QUCKSTART

1) Compile your design with vtcom as follows for use with the Undertow Suite:

vtcom -cdn_vhdl -WORK <lib_name> <file1.vhd> <file2.vhd> ...

IMPORTANT:

NOTE: By default, vtcom writes compiled files to ./ut_libs/work.

If you specify the -WORK <lib_name> option, the compiled files are written to ./ut_libs/ <lib_name> instead of ./ut_libs/work.

2) To bring up Undertow using these compiled libraries/files, use the following:

%ut -iv -use_ut_libs ./ut_libs -sigfile vt.dump -CDSLIB ./cds.lib

where, cds.lib file for example looks as follows:

DEFINE worklib ./worklib

DEFINE <lib_name> <path_to_library>

DEFINE simprim <path to simprim>

DEFINE xilinxcorelib <path to xilinxcorelib>

SOFTINCLIDE <path>/cdsvhdl.lib

The Cadence nclaunch GUI allows you to create a cds.lib file automatically without having to type in the contents manually.

3) To create a vt.dump file for your design using our VHPI, do as follows:

a) Make sure you set the following environment variables:

setenv UT_ROOT_DIR <path to UT installation directory>

setenv UT_WORK_DIR .

b) Make sure LD_LIBRARY_PATH environment variable points to $UT_ROOT_DIR/PLI which has the vtvpivhpi_nc.so file:

setenv LD_LIBRARY_PATH $UT_ROOT_DIR/PLI

c) Use the same ncvhdl & ncelab commands as before:

For example,

ncvhdl -work <lib_name> <file1.vhd> <file2.vhd> ..

ncvhdl -work simprim simprim_pkgs.vhd

ncvhdl -work xilinxcorelib blk_mem.vhd

ncvhdl -f source

ncelab worklib.top:rtl

d) The ncsim command should be as follows:

ncsim -loadvhpi vtvpivhpi_nc:vtVhpiDumpvars <root_instance>

For your example, if <root_instance> is top, add the following commands to your ncsim command line:

ncsim -loadvhpi vtvpivhpi_nc:vtVhpiDumpvars TOP

Here, TOP is the root_instance or the top level entity.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

VTCOM COMPILER

VERITOOLS COMPILER ADAPTER (vtcom)

I) USING VTCOM WITH MAKEFILE:

Users can run their makefile using the special options listed below including the specification to use vtcom instead of their own

compiler.

VTCOM creates a ut_libs directory that contains Verific translated tcl commands to load the design and search directives for the

precompiled vhdl libraries (including the design).

Supports:

MODELTECH COMPILER

-- MTI Compiler (invoke with -mti_verilog or -mti_vhdl)

MTI Makefile should be invoked like this:

make -f makefile VCOM="vtcom -mti_vhdl" VLOG="vtcom -mti_verilog"

CADENCE COMPILER

-- CDN Compiler (invoke with -cdn_verilog or -cdn_vhdl)

CDN Makefile should be invoked like this:

make -f makefile ncvhdl="vtcom -cdn_vhdl" ncvlog="vtcom -cdn_verilog"

SYNOPSYS COMPILER

-- SNPS Compiler (invoke with -snps_verilog or -snps_vhdl)

SNPS Makefile should be invoked like this:

make -f makefile vhdlan="vtcom -snps_vhdl" vcs="vtcom -snps_verilog"

Output:

- Creates a ./ut_libs directory. If UT_WORK_DIR is defined, then creates a $UT_WORK_DIR/ut_libs directory

- Creates a ./ut_libs/commands.tcl file which contains Undertow ordered-flat compiler directives (Verilog-VHDL)

- Creates a ./ut_libs/search_lib.tcl file which contains Undertow compiled libraries search paths (VHDL)

- Creates a ./ut_libs/libs file which contains Undertow ordered list of libraries for compilation (Verilog-VHDL)

- Creates a ./ut_libs/{libs}/commands.tcl file which contains Undertow ordered

per-library compiler directives (Verilog-VHDL)

- Creates precompiled standard package libraries ./ut_libs/{ieee, std, synopsys, arithmetic, qsim_logic}/*.vdb if VHDL is used

- Creates precompiled design libraries under ./ut_libs/{libs}/*.vdb if VHDL is

used

Settings:

- Environment variable UT_ROOT_DIR has to be defined.

- Environment variable UT_WORK_DIR may be defined to compile in a different directory from the current directory.

- Environment variable UT_STANDARD_VHDL may be defined to fetch user's STANDARD VHDL packages

II) USING VTCOM WITHOUT MAKEFILE:

The syntax for compiling the design using vtcom without using a makefile is as follows:

MODELTECH COMPILER

vtcom -mti_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

Default:

Verilog 2001

VHDL 93

Command line options: vtcom supports most of the command line options supported by the MTI compiler (vcom/vlog).

-2001 (for Verilog 2000)

-95 (for Verilog 95)

-93 (default VHDL 93)

-87 (for VHDL 87)

-2002 (NOT SUPPORTED)

By default, vtcom writes compiled files to ./ut_libs/work.

If you specify the -work <lib_name> option, the compiled files are written to ./ut_libs/ <lib_name> instead of ./ut_libs/work.

CADENCE COMPILER

vtcom -cdn_vhdl -WORK <lib_name> <file1.vhd> <file2.vhd> ...

Default:

Verilog 2001

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

VHDL 87

Command line options: vtcom supports most of the command line options supported by the CDN compiler (ncvhdl/ncvlog).

-V1995 (for Verilog 95)

-V93 (for VHDL 93)

By default, vtcom writes compiled files to ./ut_libs/work.

If you specify the -WORK <lib_name> option, the compiled files are written to ./ut_libs/<lib_name> instead of ./ut_libs/work.

SYNOPSYS COMPILER

vtcom -snps_vhdl -work <lib_name> <file1.vhd> <file2.vhd> ...

Default:

Verilog 95

VHDL 93

Command line options: vtcom supports most of the command line options supported by the VCS compiler (vhdlan/vcs).

+v2k (for Verilog 2000)

-vhdl87 (for VHDL 87)

By default, vtcom writes compiled files to ./ut_libs/work.

If you specify the -work <lib_name> option, the compiled files are written to ./ut_libs/<lib_name> instead of ./ut_libs/work.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Quick Start Guide: Verilog Example

Preparing for Verilog Quick Start

The example file used in this Quick Start is found in FSM directory in the Undertow distribution directory. Using this Verilog example allows the user to quickly see how the Undertow Suite enables source code debugging in a batch process. Continuing with this example also allows the user to see how to run an interactive session with a Verilog-XL simulator.

Defining Mouse Controls

The following are the definitions of the mouse operations which will apply during this Quick Start example.

- **Select or click** - use *left* mouse button.
- **Double-click** - use the *left* mouse button to rapidly click twice.
- **Right-click** - use the *right* mouse button.
- **Drag and drop** - press the *middle* mouse button over an item and while still pressing down, drag it to the new location.
- **left mouse button** - sets the T0 cursor in the Waveform Window.
- **Middle mouse button** - sets the T1 cursor and all additional cursors in the Waveform Window.

For example, mouse operations left, middle, middle sets the T0, T1 and another T1 cursor with a delta tags between the T0 cursorand the first T1 cursor and another delta tag between the first T1 cursor and the second T1 cursor

- **Box zoom** - press the *right* mouse button and slide it over the area to be enlarged, only releasing when box includes the desired area.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Setting Environment Variables

Your environment variables must be set in order for this example to run correctly.

1. Set the following environment variable to the path and directory of your Undertow distribution directory, the location where your Undertow executable is found:

   setenv UT_ROOT_DI*R* <*full_path_directory*>

2. Set the following environment variable to the path and directory of your Undertow work directory, the default location for file dialogs:

   setenv UT_WORK_DI*R* <*full_path_directory*>

3. Set the following environment variable to the path and file name of your license file:

   setenv VERITOOLS_LICENCE_FILE <*full_path_directory*>

4. For interactive portion of the demo, set the following environment variable to the path and file name of your Cadence distribution directory, the location where your Cadence executable is found:

   setenv CDS_ROOT <*full_path_directory*>

Starting Verilog Quick Start

Entering Start Command

In the FSM directory, use the command line as follows:

%ut -iv -f fsm.vc -sigfile fsm.sigs -tracefile fsm.trace

The command line above is explained as follows:

- "**ut**" is used to start the Undertow.
- "**-iv**" brings up the source code debugging window with Undertow.
- "**-f fsm.vc**" specifies the Verilog source code file, "**-f**" is followed by the name of a file holding the source code files.

 Copyright Veritools 2004 All Rights Reserved

- "**-sigfile fsm.sigs**" specifies the signal file, "**-sigfile**" is followed by the name of the waveform file. Before running Undertow Suite as a batch process, have this file already on your disk. When Undertow Suite is run as an interactive process, this file will be written out by the simulator during simulation.

- "**-tracefile fsm.trace**" specifies the trace file, "**-tracefile**" is followed by the name of the trace file. Before running Undertow Suite as a batch process, have this file already on your disk. When Undertow Suite is run as an interactive process, this file will be written out by the simulator during simulation.

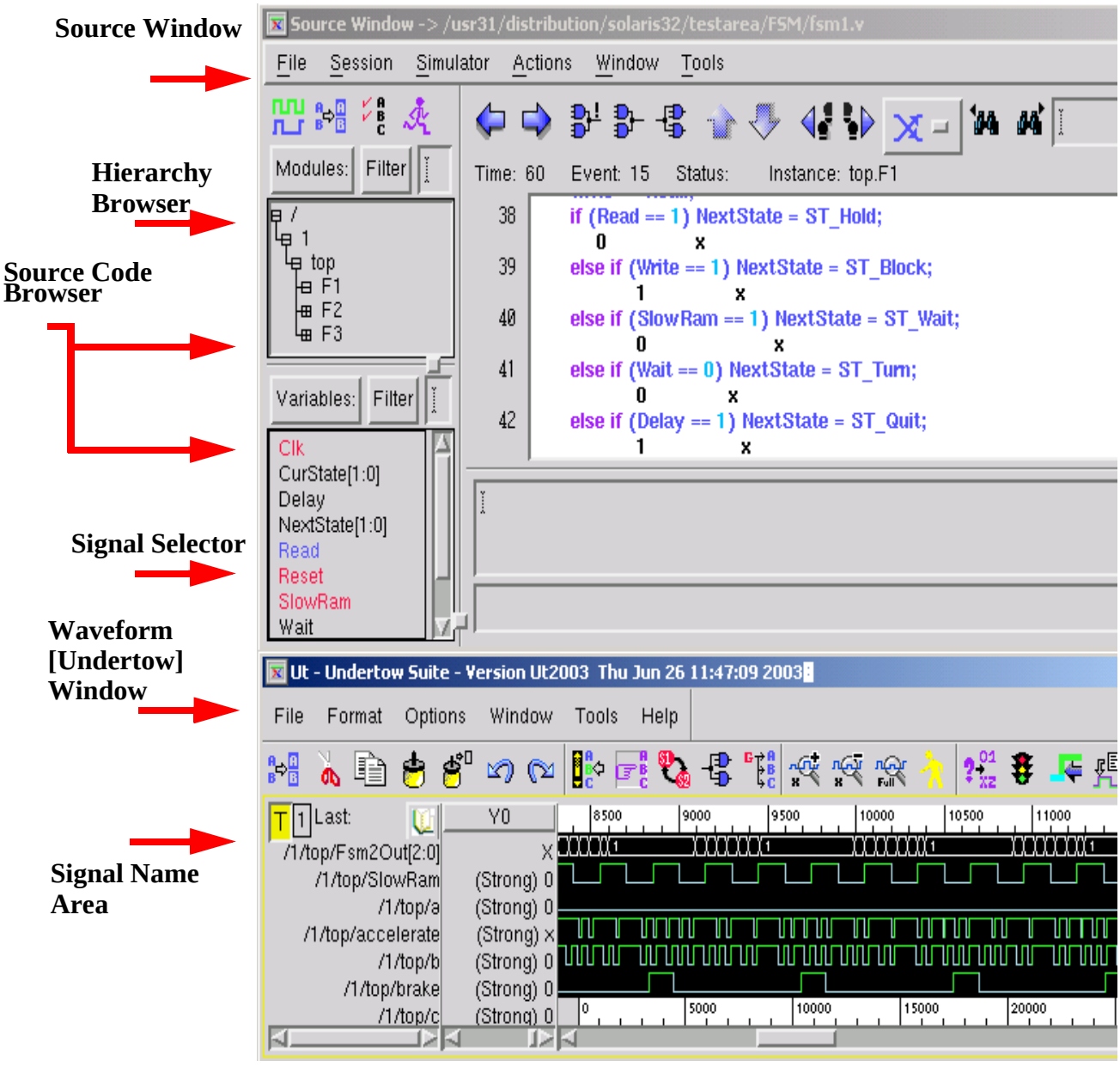
Note: A trace file will provide the users with a virtual simulator, however it will slow the simulation done by a factor of 2-4 times, so it should by used only when simulation times are not long or when virtual simualtion is required.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Viewing Undertow Suite

When the tool comes up, the windows displayed in Figure 161 appear.

FIGURE 161. Reference example for viewing the Undertow Suite windows



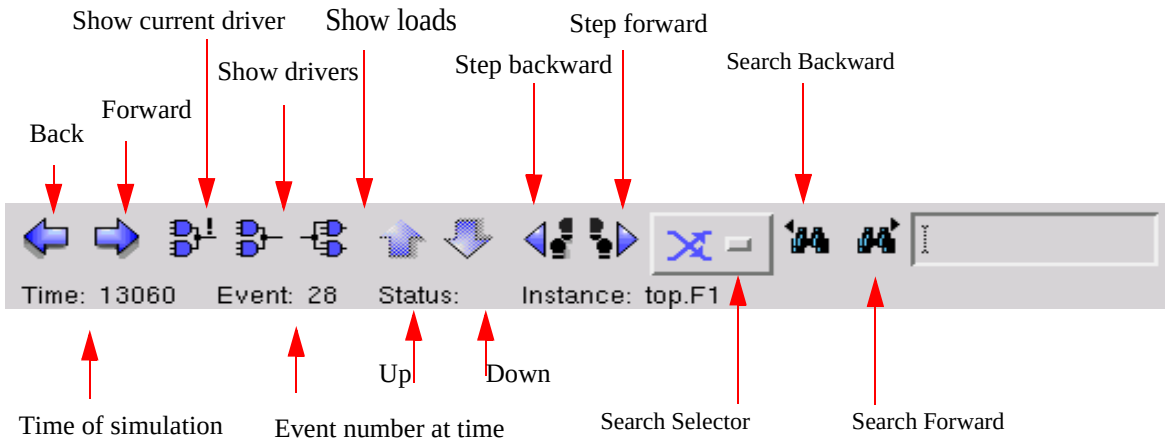
 Copyright Veritools 2004 All Rights Reserved

Source Code Window

The Source Code Window is located at the upper right side of the screen (See Figure 161). The file that is loaded into the Source Code window is shown on the header at the top of this window. When this tool is running an interactive simulation, an area located at the bottom of the Source Code Window holds the text output from the simulator, which details what operations the simulator is doing. A text input line is located just below this area, which users can use to type in simulation commands when needed.

Run the mouse cursor over any icon and the tooltips will appear as a text box displaying the icon name (See Figure 162).

FIGURE 162. Source Code Window toolbar

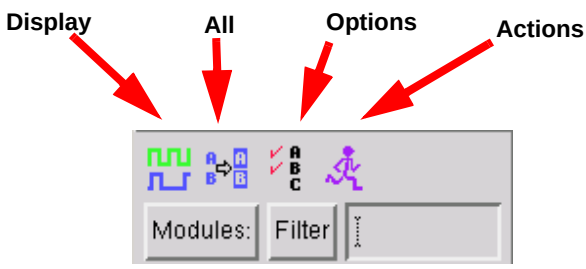


Source Code Browser

The Source Code Browser is located to the left of the Source Code Window (See Figure 161). It includes two areas or windows, the Hierarchy Browser above and the Signal Selector below.

Run the mouse cursor over any icon and the tooltips will appear as a text box displaying the icon name

FIGURE 163. Source Code Browser toolbar



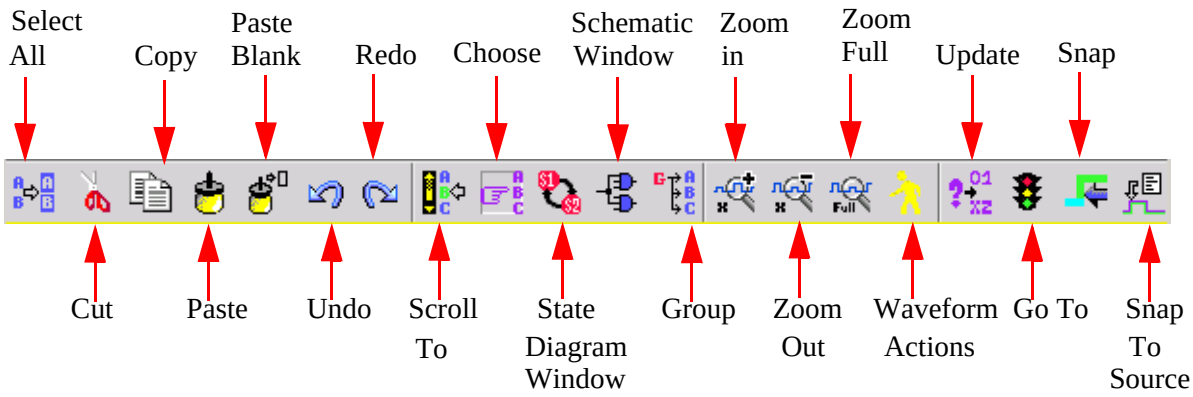
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Waveform or Undertow Window

The Waveform Window is located at the bottom area of the screen (See Figure 161). The window has three areas: the Signal Names Area on the left-hand side, the Signal Values Area is in the middle, and the Waveform Display Area on the right-hand side.

Run the mouse cursor over an icon and the tooltips will appear as a text box displaying the icon name (See Figure 164).

FIGURE 164. Waveform Window tool bar

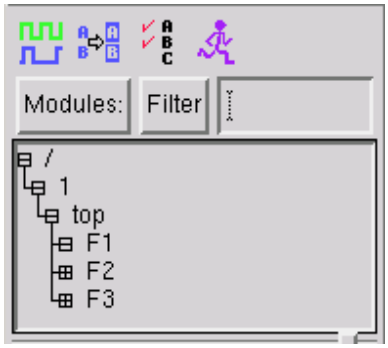


Using the Basic Features of Undertow Suite

When the Undertow Suite was started for the example used in this chapter, specific files were loaded that provide the source code for debugging.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 165. Hierarchical Browser



Using the Hierarchical Browser

To use Quick start, go to the Hierarchical Browser Window. In the Hierarchical Browser, the hierarchy is shown with a /1 at the very top, and below this the module name for the top hierarchical level of this design is displayed, in this case, the module “top”. To the immediate left of each module name is a “+” sign, indicating that this hierarchical level has not yet been expanded. When expanded or opened at this level, the “+” sign will be turned into a “-” sign and the hierarchy level one level below this level will be opened.

1. Click on the “+” sign to the left of the number 1 in the Hierarchical Browser. The next level below now displays the module name for the top hierarchical level of this design, in this case the module “top”.

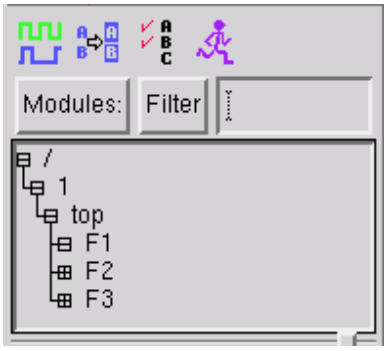
    - To expand or open the hierarchy at any level, click on the “+” sign, it will be turned in to a “-” sign and will display one hierarchy level below your current level.

    - To close the hierarchy at this level click the “-” sign (See Figure 166).

Select the folder in the hierarchy browser in order to see the corresponding source code in the Source Code window. The folder at this level will "Open" indicating that the source code at this level is displayed in the source cod window. Select the Module name at any level to see the signals displayed in the signal selector window at this level. Once the module name is selected, it will be highlighted, indicating that the signal names in the signal selector are from this level.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 166. Hierarchy view for the *fsm.vc* example

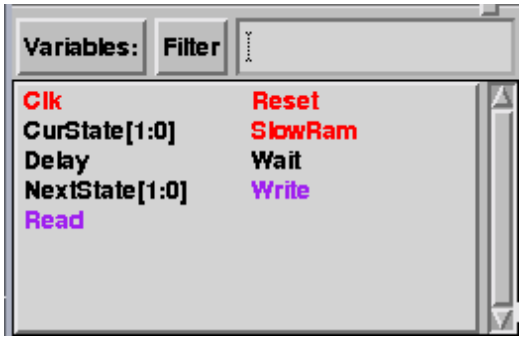


2. Click on the name at any hierarchical level as this will cause the signal names to be displayed in the Signal Selector. The module name selected will be highlighted (reverse video) indicating that the signal names of this level will be displayed in the Signal Selector.

   In the Signal Selector, the signal names will be color coded as follows:

   - **Red** - if the signals are inputs.
   - **Blue** - if the signals are outputs.
   - **Black** - if the signals are neither inputs nor outputs.
   - Purple - if inputs and outputs are combined (See Figure 167).

FIGURE 167. Signal Selector displaying signals for the F1 module



Displaying Selected Signals in the Waveform Window

To display selected signals in the Waveform Window proceed as follows:

1. Select the module at hierarchy level from the Hierarchical Browser (See Figure 161).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select signals in the Signal Selector (left mouse button). Selected signals remain highlighted with a white background until unselected by clicking the same signals for a second time. Multiple signals may be selected at one time choosing one of the following methods:

   - Individually clicking each signal.
   - Selecting the first signal desired and then dragging the cursor across contiguous signals.

3. The selected signals can be displayed by using one of the following methods:

   - Dragging and dropping (middle mouse button) the selected signals to Signal Name Area of the Waveform Window.
   - Selecting the **Display** icon on the Source Code Browser (See Figure 163). The currently selected signals will immediately be displayed in the Signal Name Area.

   Note: If no signals are selected and you press the Display icon, it will toggle down. At this point any signal that is now selected will be immediately displayed in the Waveform Window at the bottom of this window.

   - Selecting the **All** icon from the Source Code Browser (See Figure 163) will display all the signals from the highlighted hierarchical level in the Waveform Window.

   Note: If the waveform file being displayed were as large as 2 gigabytes, the speed of the display would be exactly the same as for the demo file. This is because, the speed of the waveform display is independent of the size of the waveform file, even when the waveform files are huge.

4. Select each hierarchical level, F1, F2, and F3 and then repeat the above steps in order to display signals at each of these levels.

   Note: The data in these files are compressed by the Verilog PLI routines by 20X, 40X, 80X or 300+X when writing out the waveform files from the simulator when using the Veritools' PLI routines. (See PLI section on how to link in PLI and run).

Displaying Source Code for any module

To show the instanatiation in top level: You can drag and drop (middle mouse button) any module from the Hierarchical Browser to the Source Code Window to view the instantiations of that module in the source code.

To show the content of the module: Clicking on the folder next to the module name at will display the source code contents of that module. The Source Code window and the Signal selector window will be synchronized as it will display the signal on that module.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Finding a specific Variable

To find every line of source code that includes a specific signal name proceed as follows:

1. Drag and drop the module name from the Hierarchical Browser to the Source Code Window in order to display the source code for this module (See Figure 161).

   Make sure to turn off the **Display** icon in the Source Code Browser (See Figure 163), then select the desired signal name from the Signal Selector.

2. Select the **Variable** button (at the top of the Signal Selector), to see the first line of source code which uses this variable name.

3. To see all lines of code containing this signal name, click the **Variable** button until the code line numbers in the Source Code Window begin repeating.

Virtual Simulation Feature

The Undertow Suite is capable of doing virtual simulation if it has loaded a trace file.

The command line used to load Undertow Suite for this example included a trace file name "**fsm.trace**".

*Note: On the Source Code Window press **Simulator => Kill** to verify that the Undertow Suite is not actively running a simulator. A dialog box will display "The Simulator is not running". Dismiss the dialog box.*

1. Press the **Steps Forward** icon on the Source Code Window (See Figure 162). This steps the virtual simulation forward one event.

2. Press the **Steps Backward** icon on the Source Code Window (See Figure 162). This will step the virtual simulation backward one event.

3. Repeat Steps 1 and 2 while noting the changes. Notice that the time point and event number are located just below the icons but above the source code in the Source Code Window.

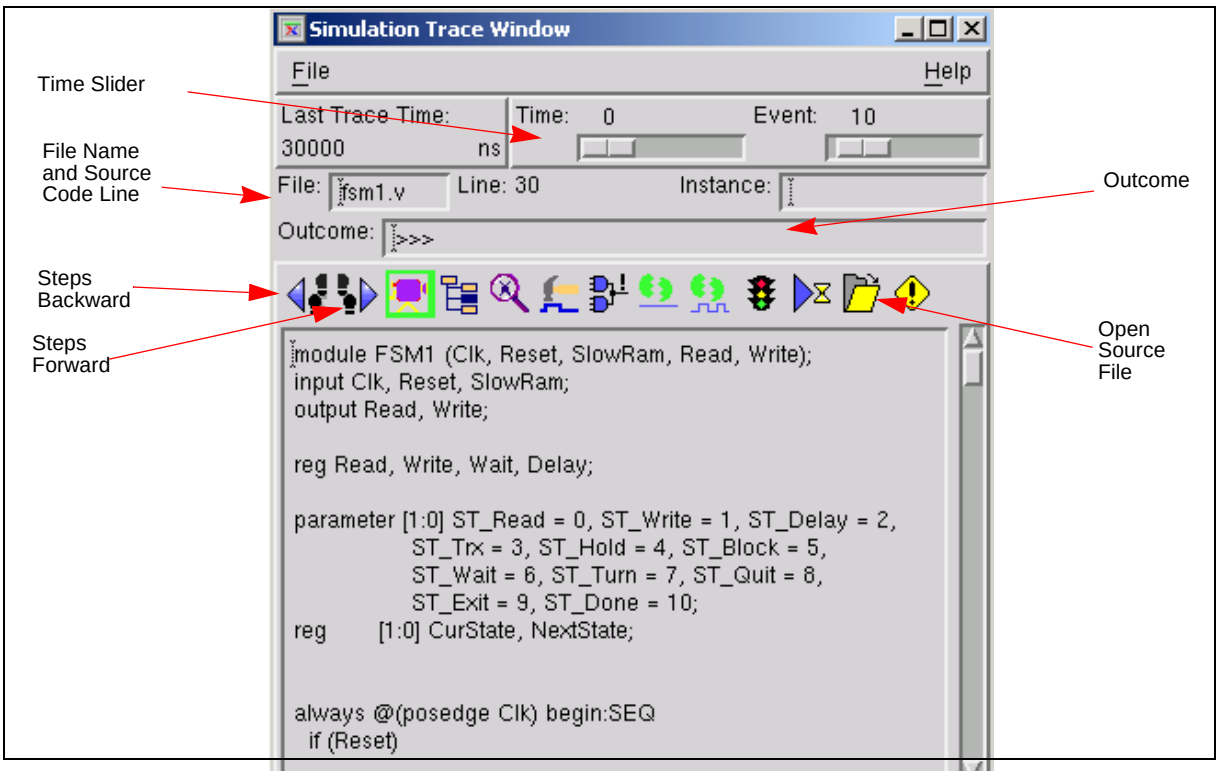
4. Open the Trace Window by choosing the following from the Source Code Window menu:

   Tools =>Trace.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Note that the Trace Window has a time slider (See Figure 168). This time slider allows the user to access any time point in the simulation. Slide this slider to any location and allow the Trace Window to update.

FIGURE 168. Simulation Trace Window



5. Press the **Steps Forward** or the **Steps Backward** icons in the Trace Window to step through the simulation events in the Trace Window in the exact sequence of the simulation execution that has been run in batch mode, and is now saved in the trace file.

   The file name, line in that file, and the actual outcome of each line of code that is executed are shown above the icon bar in the Trace Window.

6. Press the **Open Source File** icon to open, or bring to the top, the Source Code Window and highlight the last line of code that you were on when stepping the Trace Window.

7. To close the Trace Window select *File =>Close.*

Snap to Source

Snap to source provides a convenient way to view the line of source code in the Source Code Window that had caused any transition on any signal in the Waveform Window.

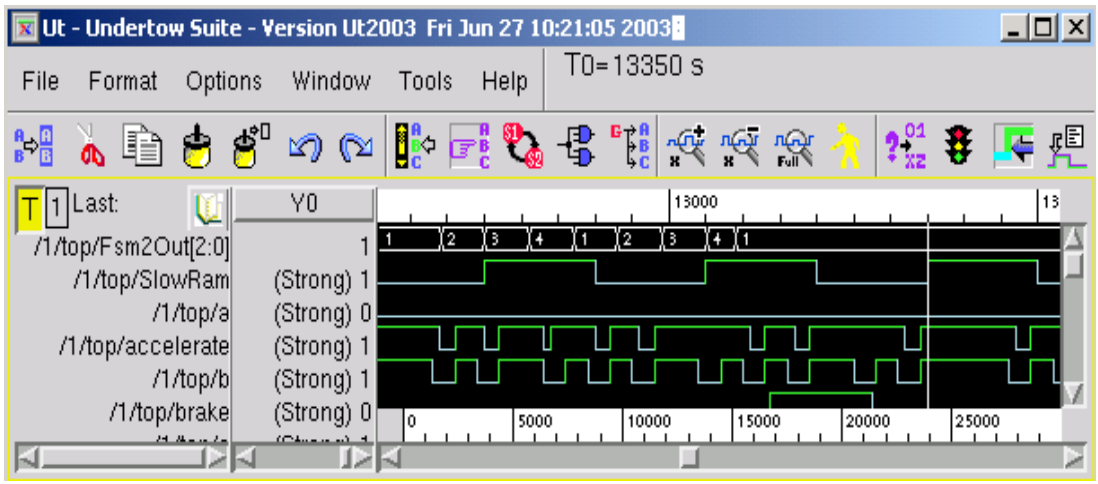
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To use Snap to Source, proceed as follows:

1. Zoom into a signal in the Waveform Display Area. Use the right mouse button to press and slide over the desired area to zoom in on.

   The zoomed area should allow you to easily see the edges and pulses of the displayed waveforms (See Figure 169).

FIGURE 169. Snapping to a signal edge



2. Press the **Snap** icon to turn snap on in the Waveform Window (See Figure 164).

3. Snap to an edge of any signal with T0 cursor using the left mouse button.

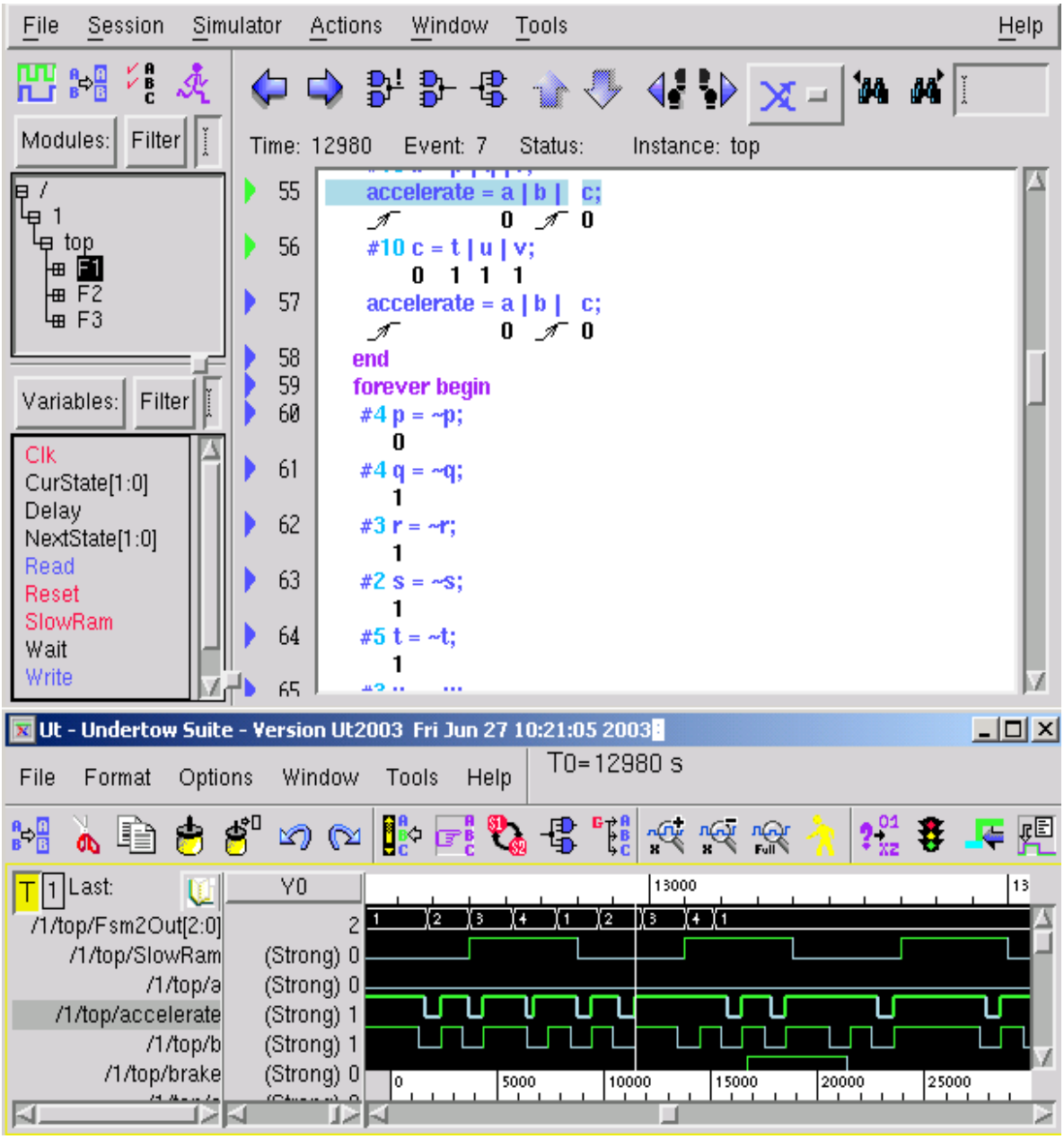
   The edge you will snap to is defined to be the edge on a signal where the cursor arrow tip lies between this signal's one and zero level, and is the edge when snapping in a signal pulse that is the closest edge to the snap cursor. The snap operation will snap the T0 cursor to that edge and places the time for the T0 cursor above the icons in the Waveform Window. Also note, that this operation, as will any snap operation, causes the annotated values in the Source Code Window to update to the signal values at this timepoint. The annotated values are those values displayed just below each variable in the Source Code Window.

4. Next press the **Snap/Source** icon to turn Snap to Source on (See Figure 164), this is the icon to the immediate right of the Snap icon.

 Copyright Veritools 2004 All Rights Reserved

5. Now, snap the T0 cursor to the edge of any signal, using the left mouse button, signal */(1)/top accelerate* is used for this example (See Figure 170).

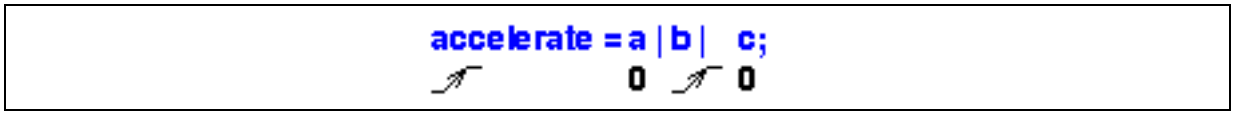
FIGURE 170. Snapping to source



6. The Source Code Window will now display and highlight the exact line of code that had caused the edge on this signal to transition. Notice that the line of source code for the edge in this example is line # 55. The exact line of source code depends on which edge you have

selected. The Verilog expression on this line of Verilog source code is displayed in Figure 171.

FIGURE 171. Viewing line # 55 source code example



Also notice from Figure 171, that the signals are annotated with their values at the time of the T0 cursor. The signal *accelerate* is annotated with an annotation symbol that indicates that this signal edge is transitioning from a zero value to a one value,. This is a symbol showing a arrow at a 45% angle going left to right, coming from a zero level, and going up to a one level.

Notice that at this point:

a = 0, b = transition from a zero to a one, c = 0

Hence it is the transition on “b” that is causing the transition on *accelerate*.

Note the Snap to Source will show the exact line of source code for any signal edge if you have loaded a Trace file.

If you have not loaded a trace file , it will display all possible lines of code and assignments that could have driven this signal, in the Details window at the bottom of the Source Code window. Use the Up/Down Arrow icons to see where all of these lines are located in the source code.

Tracing Back the Drivers to Their Ultimate Source

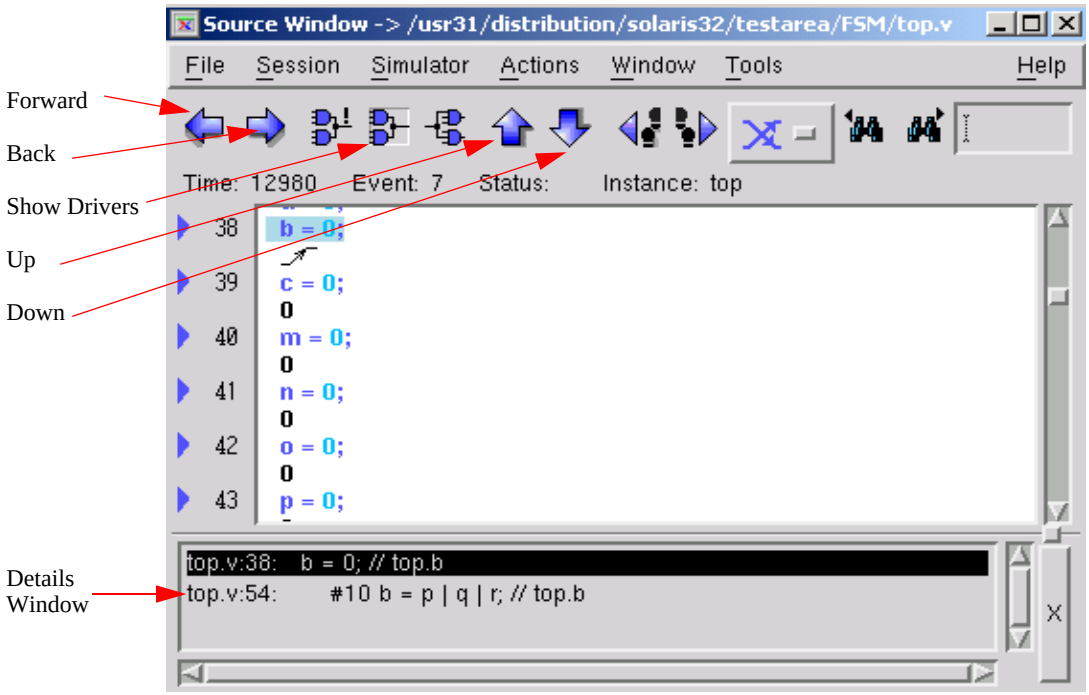
In order to trace back the drivers on the right side of the Verilog expression *accelerate =a|b|c*, we do the following steps:

1. Select **Show Driver** icon on Source Code Window (See Figure 162).

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select the signal “b” from the code. All the drivers for “b” are now displayed in the Details Window found at the bottom of the Source Code Window (See Figure 172).

FIGURE 172. Tracing back drivers



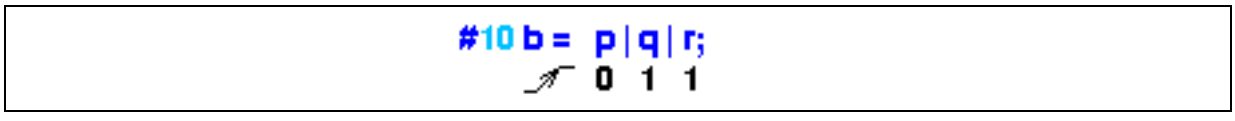
Note that the following two drivers or assignments are displayed for the signal “b”:

b = 0

b = p|q|r

3. To find the correct assignment for “b”, use the **Up** or **Down** icon to select the assignment that gives the correct value for “b” at this time point. The correct assignment for the value “b” in this case is displayed in Figure 173.

FIGURE 173. Viewing line # 54 source code example



Now note that at least one of the values for the variables (p, q, or r) is equal to a one value. Now repeat this process for the signal “r” to ultimately find the exact line of code that caused the transition on *accelerate*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

4. To replay this process in reverse on the Source Code Window, click the **Back** icon to return back through these lines of source code to the original line of source code line for *accelerate* where we began this example.

This concludes the section on the quick start example demonstrating the basic features of Undertow Suite.

Using the Advanced Features of Undertow Suite

The more advanced features that are provided in Undertow Suite are: State Diagram Window, Schematic Window, and HDLLint Report Window.

Displaying Specific Windows

To display the additional windows that are a part of Undertow Suite, choose one of the following methods:

The **Tools** menu in the Source Code Window lists the window names: Schematic, State Diagram, or the HDLLint Report window. To bring up the Schematic Window and State Diagram Window, do the following:

1. On the Source Code Window select *Tools => Schematic.*
2. Then select *Tools => State Diagram.*
3. To make rearranging of these windows more orderly and easier to use, use the tile windows function found in the Windows menu on the Source Code Window:

   Window =>Tile Windows=>States and Schematic

   This keeps the windows up that are already displayed on the screen but then adds in these new windows for the State Diagram view and the Schematic view. The State Diagram Window will be displayed in the upper right hand corner of your screen, and just below it the Schematic Window will appear along with the other windows that were up.

You can do just Step 3 to bring up the Schematic and State Diagram Windows, tiled with the other windows.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

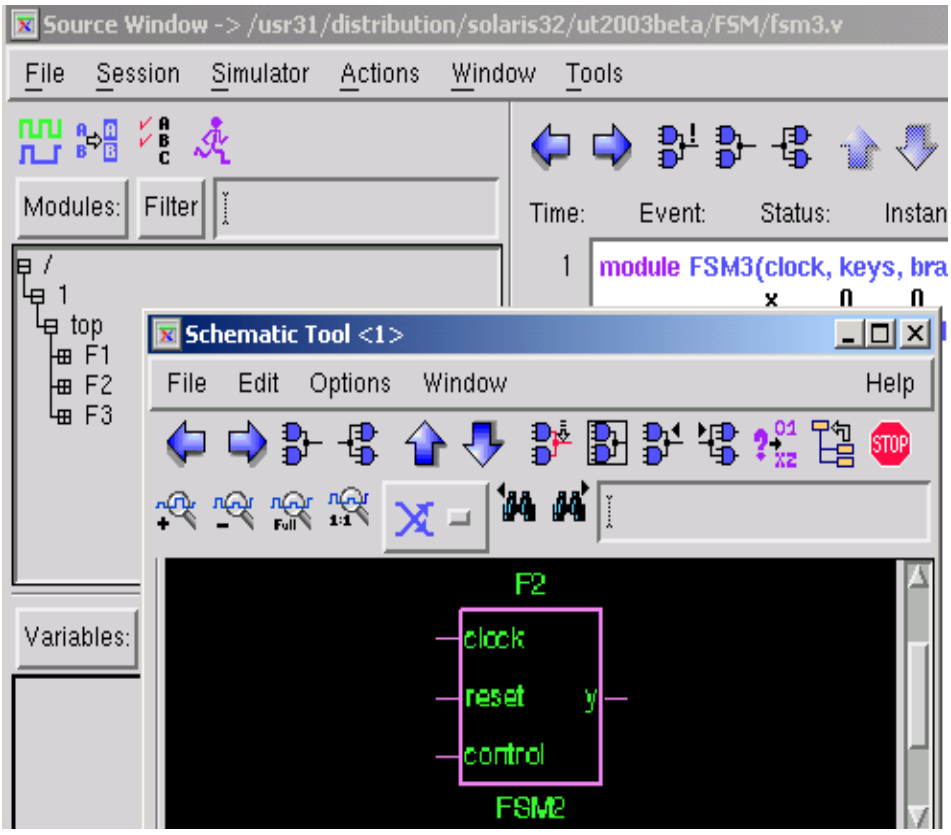
Schematic Tutorial

Bringing up the Schematic Window

In order to bring up the schematic window, press the schematic icon, in the Waveform window, and then drag and drop any module from the hierarchy browser, or any signal from the signal selector, source code window or waveform window onto this schematic window. Users can also just press the right mouse button on any module name in the hierarchy window, and the schematic window will come up with this module already displayed.

Press the right mouse button on the F2 module in the hierarchy browser to see this module in the schematic. Figure 174 shows F2 module.

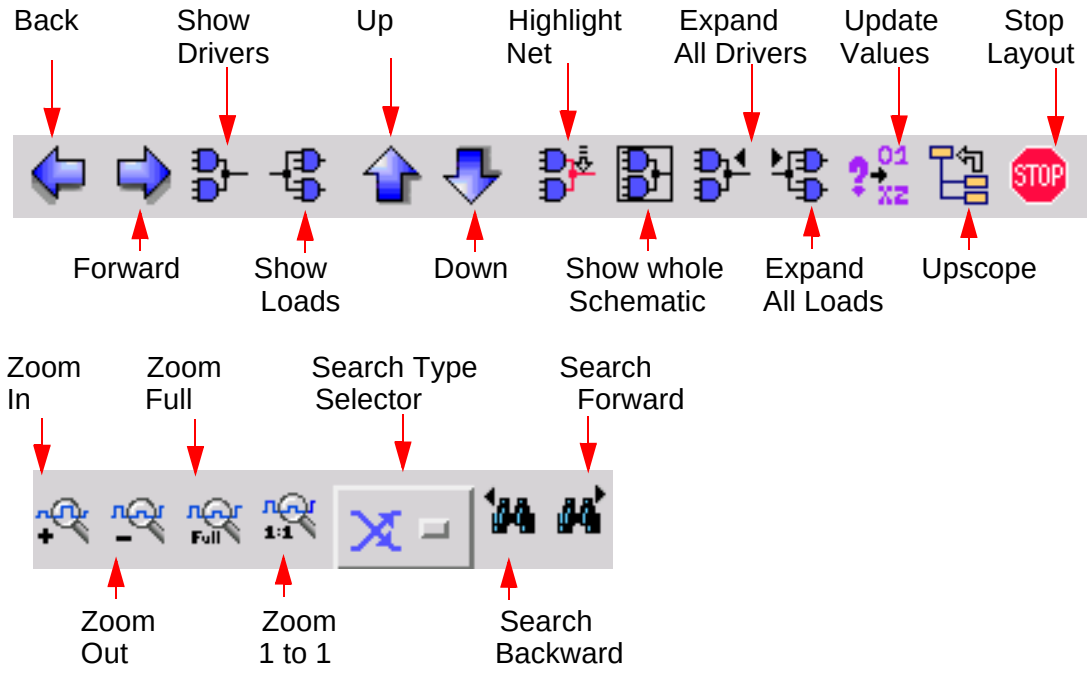
FIGURE 174. Schematic Window displaying module *F2*



Schematic Window Toolbar

Figure 175 show the icons on the Schematic Window toolbar.

FIGURE 175. Schematic Window toolbar description



- Forward - To go to the next view in the schematic.
- Back - To go back to the previous view in the schematic
- Show Drivers - expand the driver of the existing instance
- Show Loads - expand the load of the existing instance
- Up - go up one schematic that you were at.
- Down - go down
- Highlight - highlight the selected net.The net selected will be highlighted throughout the whole schematic in the scope
- Show Whole Schematic - show the whole schematic. It will display the next level up and show all elements connected to the current schematic elements.
- Expand All Drivers - expand all the driver to the selcted instance
- Expand All Loads - expands all the load to the selected instance
- Update value - updates the value of the load

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

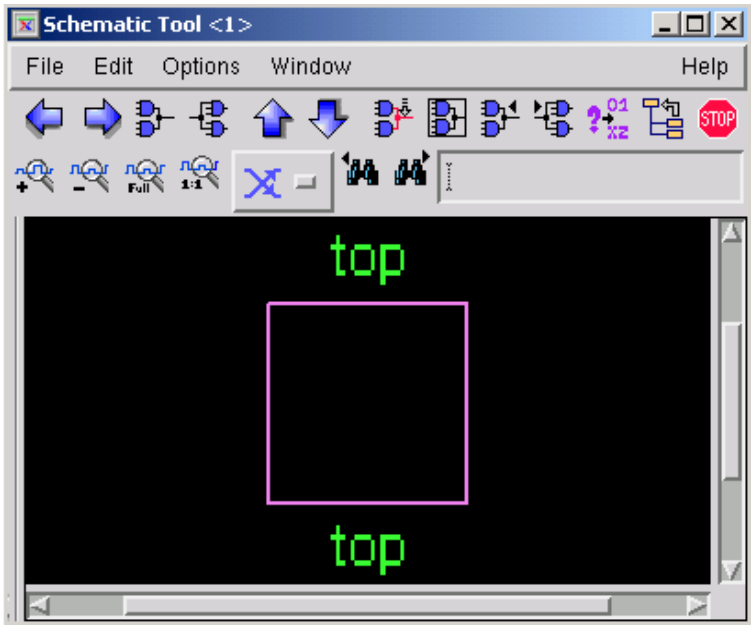
- Upscope - pressing this button will display the very next level up from the level you are currently at.
- Stop Layout - stop the display of the layout if user wants to.

Using drag and drop, the user can drag any variable from the Source Code Window, any module from the Hierarchical Browser, any signal from the Signal Selector, or any signal from the signal name list in the Waveform Window and drop this into to the Schematic window.

Note: Dragging a variable name will display the wire and all drivers for this signal connected to the left side of the signal, with all loads connected to the right side of signal. Dragging a module will display the module, with the inputs to the module shown on left side of the module and the outputs shown on right side of the module.

Drag the module names to the Schematic Window, one at a time. Drag the module *top* first (See Figure 176), followed by *F1*, *F2* and *F3*.

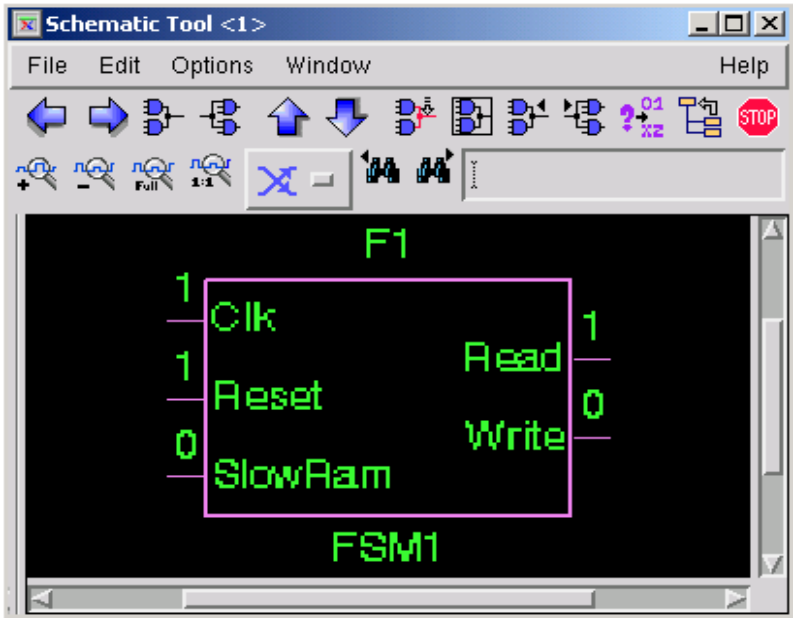
FIGURE 176. Schematic Window displaying example *top*



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Notice that on each module, the input and output pins are shown. All inputs or loads are shown on the left side of each module, the outputs or drivers are shown on the right (See Figure 177).

FIGURE 177. Schematic Window displaying example module *FSM1*



With the Schematic Window we can do the following operations:

Trace Back Logic

Logic can be individually traced backward or forward on any signal by double-clicking on its pin name, where it is connected to the element the signal is attached to (See Figure 178).

On any input pin, double-clicking once adds the wire connection and the elements that are driving this wire. Double clicking again removes this wire and all of the drivers that were previously displayed. Double clicking on any output pin does the same thing showing the wire and all of the loads connected to this wire.

To view an example of tracing back logic proceed as follows:

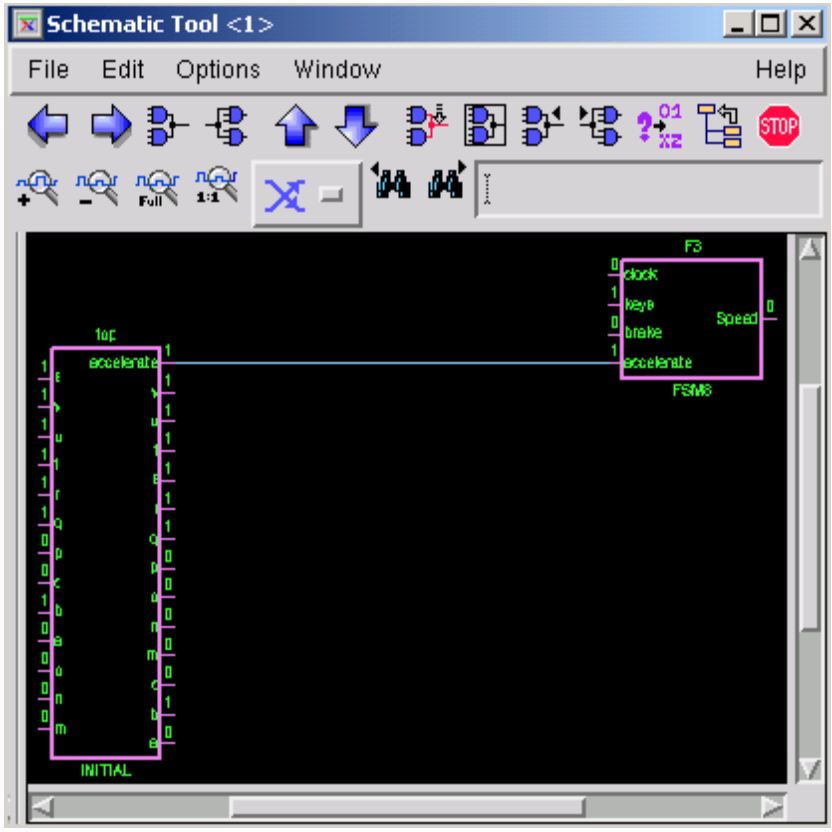
1. Drag and drop the *FSM3* element into the Schematic Window.
2. In the *FSM3* module double click the *accelerate* pin, the driving element and wire for that pin show in the Schematic Window (See Figure 178).
3. Again double click the *accelerate* pin, the driving element and wire are removed from view.

 Copyright Veritools 2004 All Rights Reserved

4. Double click *speed* pin, the load element and wire for that pin show in the Schematic Window.

5. Again double click the *speed* pin, the load element and wire are removed from view.

FIGURE 178. Tracing back logic on module *FSM3*



Displaying Drivers and Loads

When any schematic view is shown on the schematic window, users can trace the drivers for any individual connection either forward, showing the load or loads, backward, displaying the driver or drivers. To do this, double click with the left mouse button on any pin. Note that the pin highlights when the cursor is in position to select this pin. Users can also display all connection going forward or to the right, and backward or to the left. Just press, "**Expand Drivers**" to expand all of the drivers at the next level back in this schematic, press "**Expand Loads**" to expand all of the loads at the next level forward or to the right in this schematic.

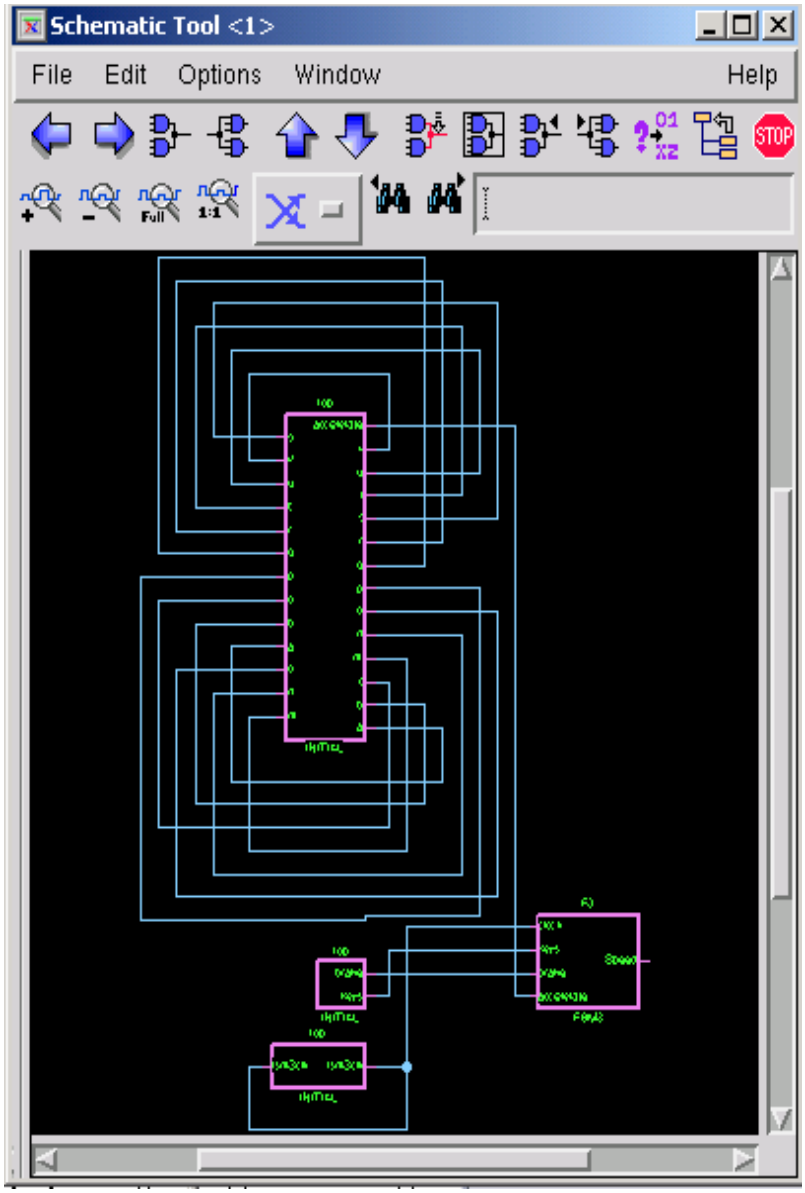
Expand All Drivers show all drivers that are connected to any unconnected load pin for the schematic level that is the level that is farthest to the left. This feature allows the user to make a trace cone or cone of logic.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To view an example of Expanding All Drivers and Expanding All Loads proceed as follows:

1. Drag and drop the FSM3 element into the Schematic Window.
2. Press the **Expand All Drivers** icon. The driving elements and wires for one more level show in the Schematic Window (See Figure 179). This can be repeated to show additional levels.
3. Press the **Expand All Loads** icon. The load elements and wires for one more level show in the Schematic Window. This can be repeated to show additional levels.

FIGURE 179. Expanding All Drivers on module *FSM3*



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

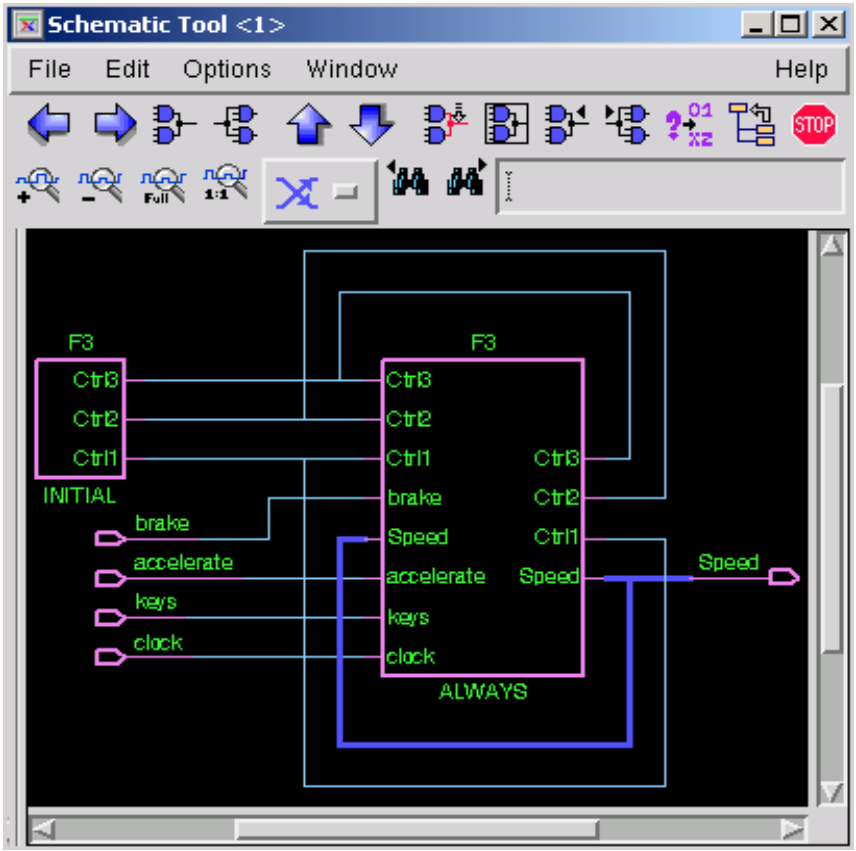
Descending the design hierarchy

In order to descend the design hierarchy when a module or block is displayed in the schematic window, double click on the empty space in the module or block. Double click in the displayed F2 Module in order to see the blocks inside this module. Now double click again on the always blocks shown in the schematic window in order to descend into this blocks. Use the icon “Back” at anytime to go back to any of the views that you had just come from. When descending into a block, such as an always block, initial block or a continuous assignment block, the Schematic window uses the HDLAnalyzer in order to synthesize the RTL logic into RTL gates.

Note that three different ways are provided in order to descend into a design hierarchy and are as follows;

1. Double clicking in this module or block shows all of the schematic at the next lower level. (See Figure 180).

FIGURE 180. Descending through the hierarchy of *FSM3 one level*



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Selecting any input or output pin, and then double clicking on the module or block will show all of the schematic for the logic connected to this pin in this module or block.

3. Double clicking on any pin shows just the logic connected to this pin, and stops after the first connected elements are displayed.

Ascending up the hierarchy

Pressing the icon, "**Upscope**", will display the very next level up from the level you are currently at, with the schematic drawing stopping at the pin boundaries for the next element.

Pressing the icon, "**Show Whole schematic**", will display the next level up and show all elements connected to the current schematic elements. All module elements are displayed as the blocks inside of these modules.

With the F2 logic shown as RTL gates, press the "**Upscope**" icon to see the elements one level up.

Then press the "**Back**" icon on the schematic window in order to get back to the synthesized F2 schematic and then press "**Show Whole Schematic**" to see all of the logic and elements at the next level up. Press "**Show Whole Schematic**" again to display all of the blocks at this level. This will display a block view of your entire design, since at this point you are at the top of this design. Note you can do any of these operations with the menu item, **Show Maximum Details** toggled on in order to see the schematic view with RTL gates replacing the schematic blocks. To set Show Maximum details on do the following;

Menu Window => Show Maximum Details => turn on

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

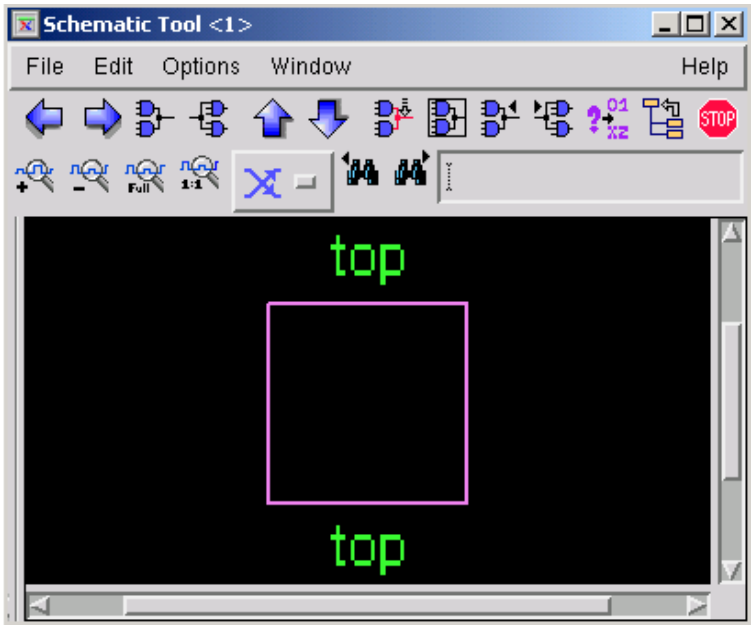
Show Whole Schematic

Any time you have any schematic image that shows input or output pins, you can press "**Show Whole Schematic**" icon to see the entire schematic.

To view a whole schematic proceed as follows:

1. Drag the module *top* on the Schematic Window (See Figure 181).

FIGURE 181. Dragging and dropping *top* on Schematic Window



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Double click into the *top* module. At this point you should see the schematic one level down (See Figure 182).

FIGURE 182. After double-clicking into *top*

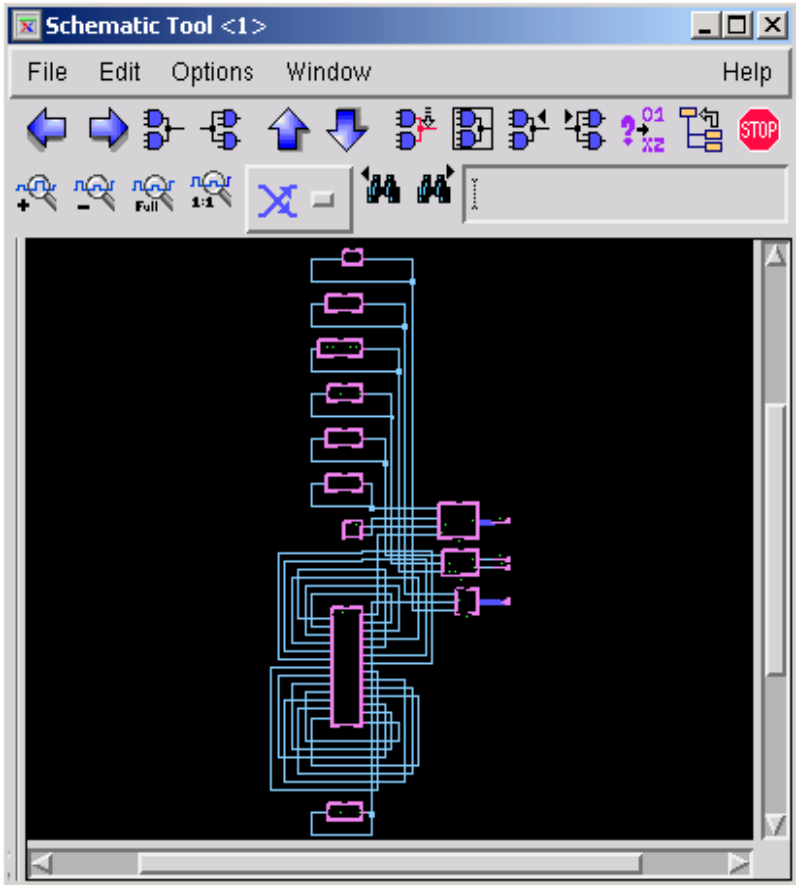
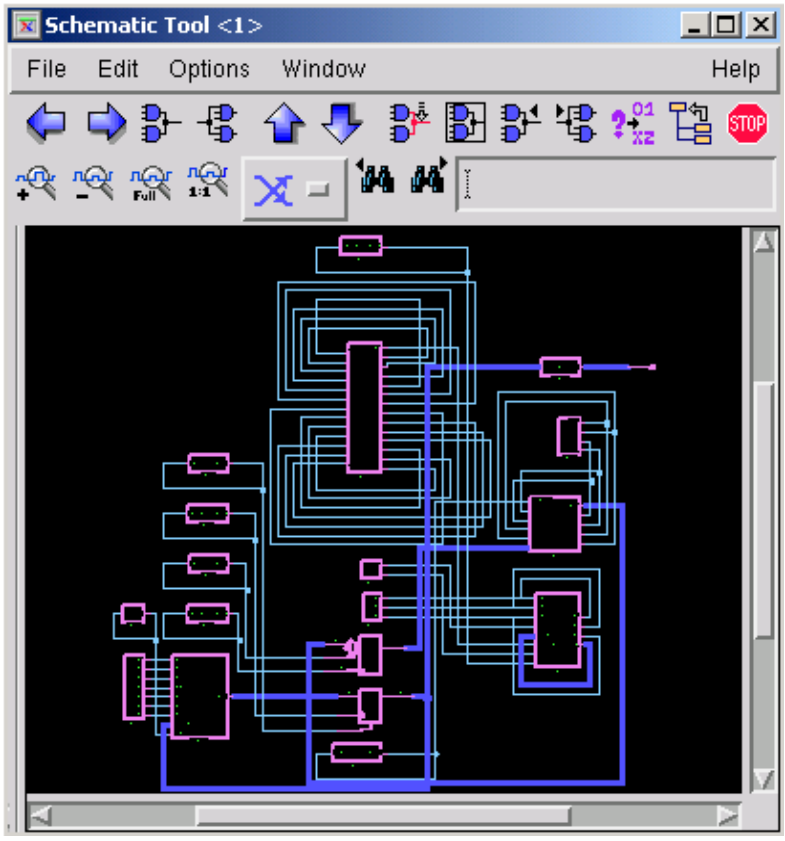


FIGURE 183. Schematic view of top module.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

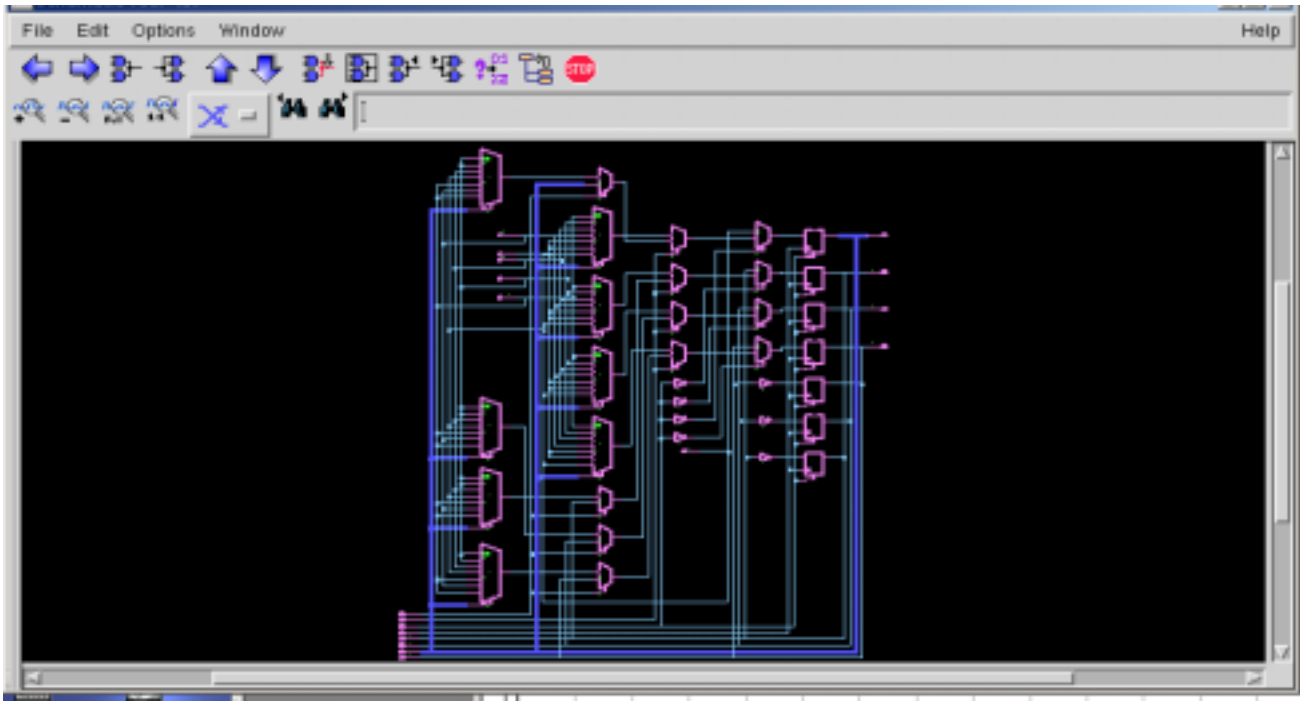
3. Now Press the **Show Whole Schematic** icon to see all of the schematic at all hierarchcal levels

FIGURE 184. After pressing "Show Whole Schematic" icon for *top*



4. **Options->Show Maximum Details**
5. Now click into the block FSM2(located at the lower right corner)
6. The synthesis view of the module will appear as shown below

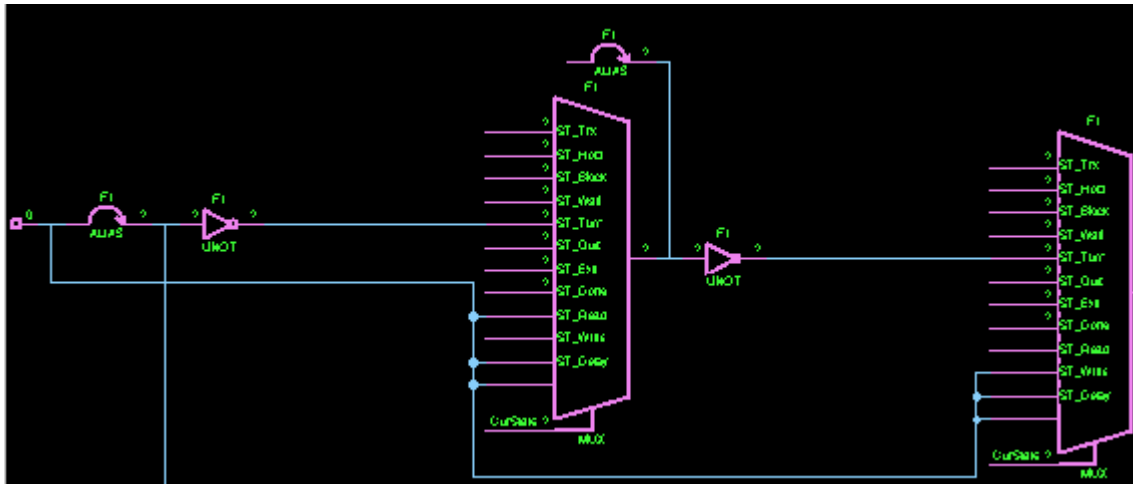
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Logic Trace Back

Logic Trace back provides a powerful tool for designer to trace back the logic path

1. On the gate level schematic, use the middle button to drag and drop any instance
2. The instance will appear on the schematic
3. Select **Option->Logic Back-trace**, and click on the input pins of the instance. The driver cell will appear.
4. Repeat it on the instance that just expanded. The path will appear as followed:



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Highlight Net

Pressing the icon, "**Highlight net**", will allow the users to display a highlighted net in the current schematic. Users can either select the net, with the left mouse button (the net/pin will highlight or flash when the cursor is in position to be selected) or can drag any of the nets from any other window in order to see that net highlighted in the current schematic. When any net is selected, or dropped onto the schematic window, that net is highlighted without removing the highlight from other nets. The last net that is highlighted can be traversed using the "**Up**" or "**Down**" icons on the schematic window.

In the schematic view of the whole FSM design, press "**Highlight Net**" on, then drag and drop signals from the Waveform window, the Signal Selector and the Source Code window with the middle mouse button to see these highlighted in this schematic view. Drag the signal fsm2clk to the schematic window from the signal selector. Note press module FSM2 to see in the Signal Selector all signals in module FSM2. Zoom in on the signal at pin, point where the signal connects to any block. Push Up/Down now to see the connections in the Schematic window.

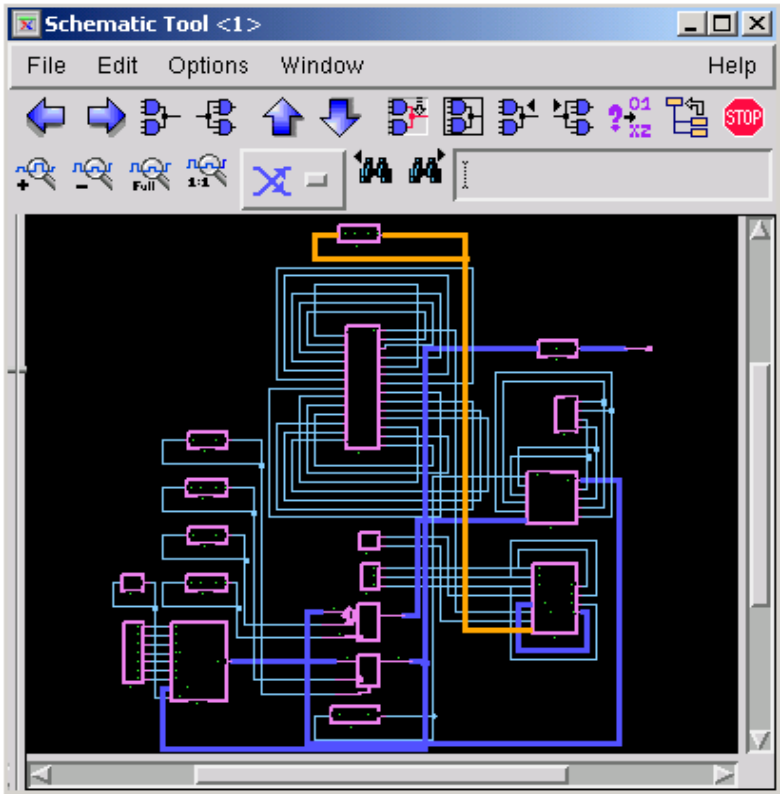
Notice also, that signal name changes depending on the hierarchy level of the signal connection.

1. After you have displayed the entire schematic from the Show Whole Schematic example, press down the **Highlight Net** icon.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select a signal from the Source Code Window, *fsm3clk* in this example, and drop this signal onto the Schematic Window. Notice that this signal is highlighted in an orange color (See Figure 185).

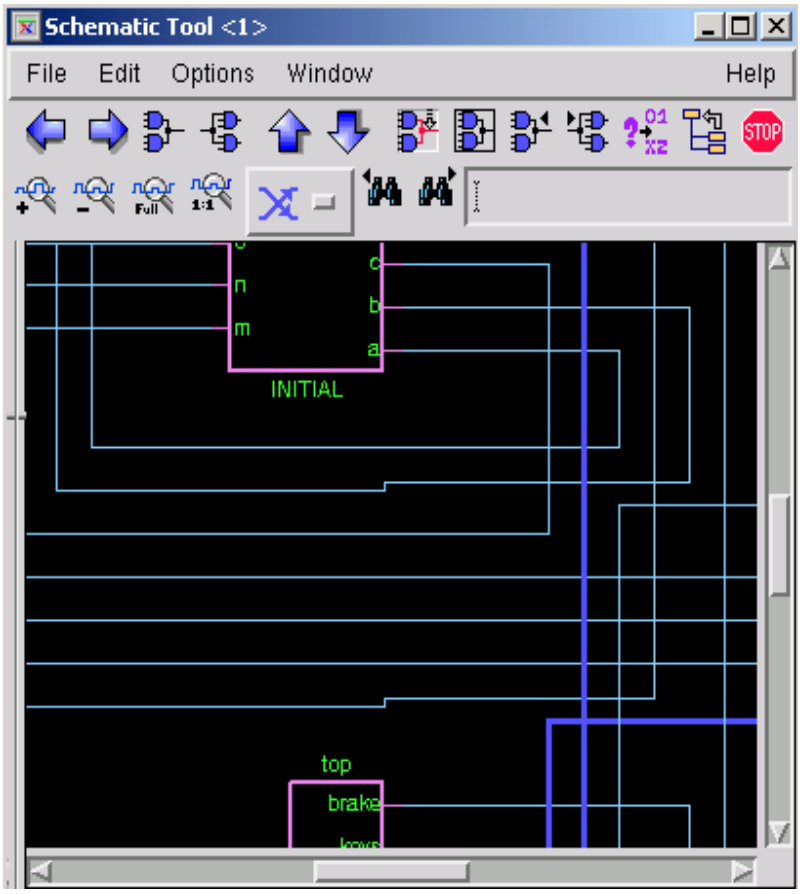
FIGURE 185. Signal highlighted in orange



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Now press the **1:1** icon to zoom into the schematic so that you can see the schematic details (See Figure 186).

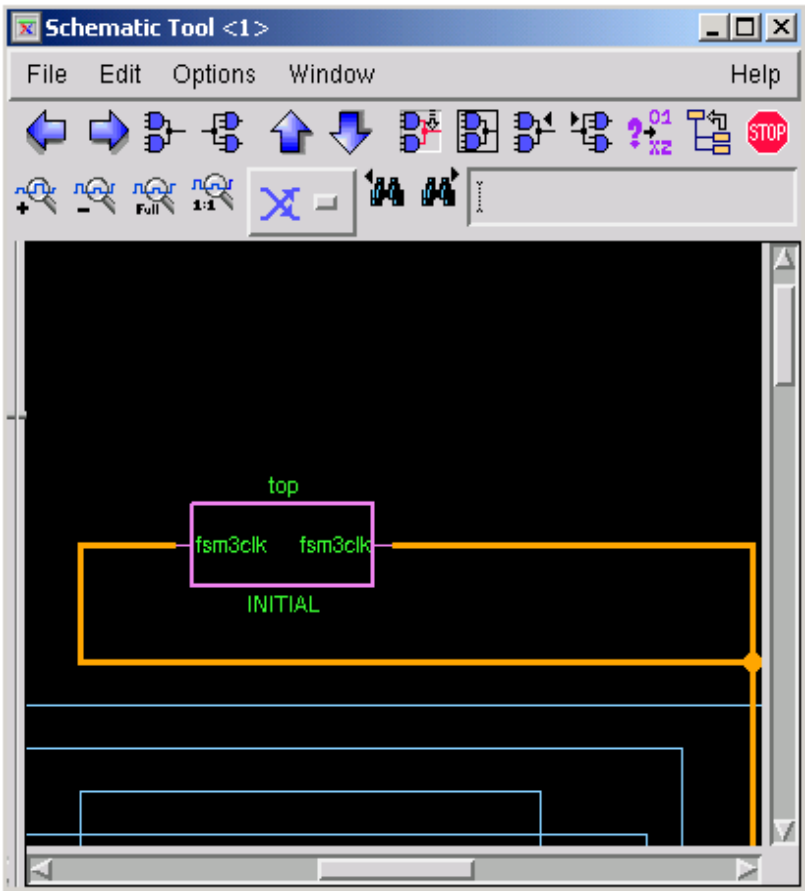
FIGURE 186. Signal zoomed 1:1 to show details



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

4. Now press the **Up** and the **Down** icons to see the connections to this net (See Figure 187).

FIGURE 187. Zoomed view moved down



Note that you can press the **Up** or **Down** icons until the schematic no longer shows a different view in order to traverse this net to the end of this wire (signal).

Viewing values annotated right on the schematic

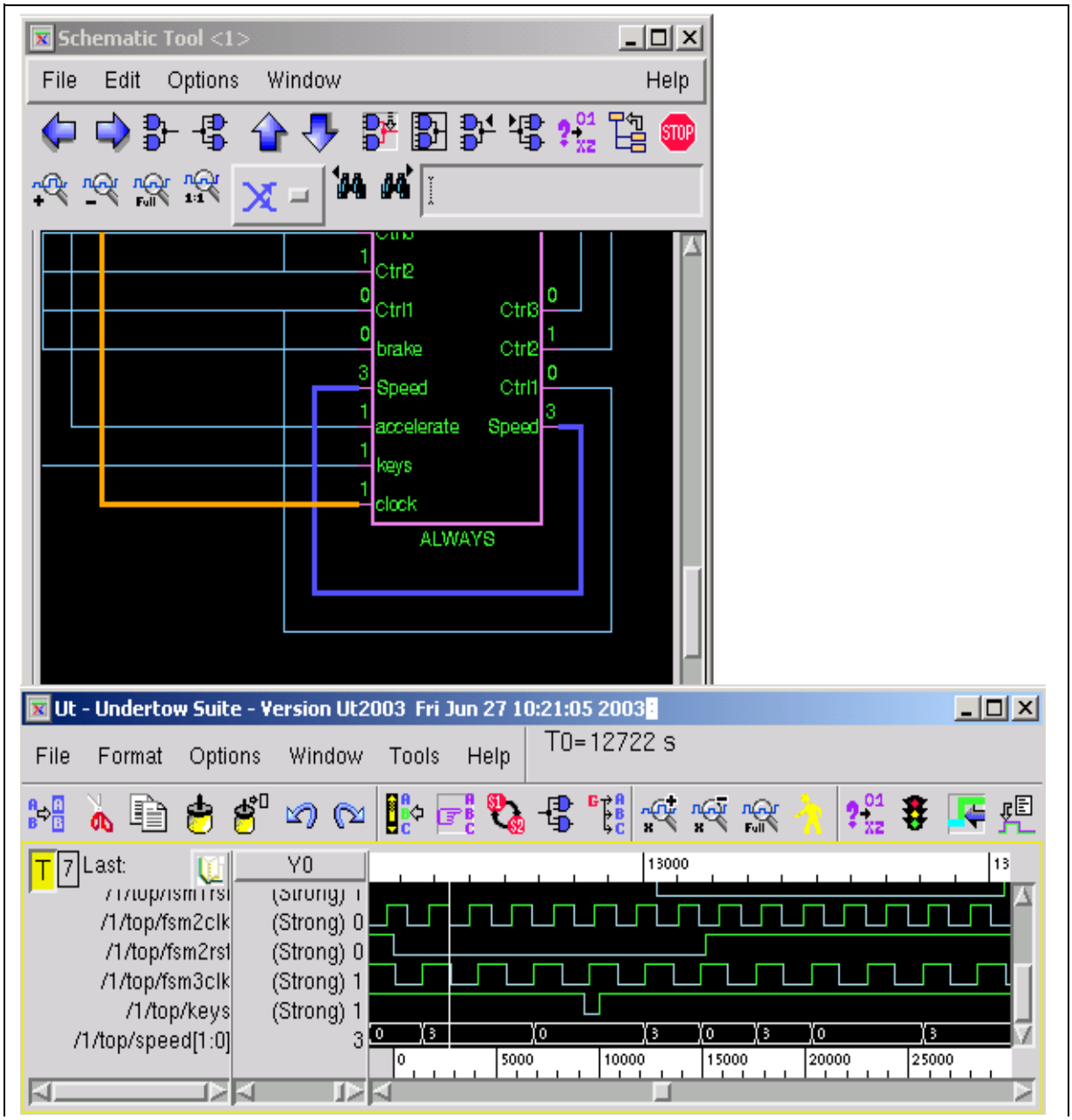
In order to view all of the values from the users waveform file annotated directly onto the schematic, press the icon "**Update Values**". This will annotate the values of the waveform file for the time position of the T0 cursor on the waveform window. As this cursor is moved to any point, the values shown in the schematic window will automatically update to show the values in the waveform file at the new time point of the T0 cursor. Users can of course drag any signal from the

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

schematic to the waveform window at any time, to see the entire waveform for this signal instead of a single value for this signal.

1. Go to the Waveform Window and place the T0 cursor anywhere in the Waveform Display Area. The schematic values will now change to the values at time for this T0 cursor.
2. Now move the T0 cursor to a new point in the Waveform Window. See that the schematic values change to reflect the values at this new time for T0 (See Figure 188).

FIGURE 188. Schematic value annotation of *NextState*



Search Forward/backward on Signal in Schematic

By selecting signal on the Schematic Window, the user can search forward or search backward to find leading edges, trailing edges, and transitions on this signal.

1. Select the clock signal on the schematic.
2. Press the **Search Type Selector** down to see the different types of searches you can do, and select **Transition** (See Figure 175).
3. Press **Search Forward** to see the value on the clock transition to the next clock edge, at the next later time in the simulation run.
4. Repeat for **Search Backward**.

Viewing the Source Code for Any Schematic Element or Wire.

The user can drag and drop any block element in the Schematic Window to the Source Code Window, in order to see the source code for this element.

The block elements include: Always Blocks, Continuous Assignment Blocks, FFs, and Initial Blocks. The user can also find all of the assignments for any wire or bus shown on the Schematic Window, by dragging and dropping this signal element from the Schematic Window to the Source Code Window.

1. Drag and drop a block element from the Schematic Window to the Source Code Window.
2. Drag and drop a signal element from the Schematic Window to the Source Code Window.

Zoom

To zoom into the Schematic Window do one of the following operations:

1. Use the right button down, slide the box over area to be zoomed and release.
2. Use zoom icons to zoom in, zoom out, full zoom and do a 1:1 zoom.

Synchronizing with the source code window

Note: As the schematic window is traversing the net and displaying every connection, with these connection points displayed in the exact center of the schematic window each time the Up/Down icons are pressed, the Source Code window by default has been set to synchronize with the schematic window. It will also display the exact point for each connection, with each point highlighted on the source code window. The line number for this connection point in the source code is also listed at the top of the source code window

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Multiple Schematic windows

Users can add more schematic window, or force the tool to show just one single additional schematic window when traversing through the schematic for a design. To get the additional windows do the following;

Window => Create New Schematic, or Window => Create One New Schematic

Drag and drop any element for any other window in the tool onto this new schematic window in order to see the schematic displayed now in this new window. This new window is particularly useful when users are tracing logic in one window and the schematic gets too complex. They can either drag and drop a single element to a new schematic or drag and drop any element to their current schematic view to limit the amount of logic they are looking at.

In the FSM design, Click **Create New Schematic**, drag the FSM block into this window, and double click into this module, and then the FSM always block to see at the same time, the whole schematic and the detailed design of this module. Now highlight the output Next state in the detailed view, and see that this signal will be highlighted in both schematic views. This is very useful when synchronizing between a synthesized view and the higher-level block diagram view.

The Schematic Window will work equally well on pure RTL source code, gate code or RTL source code mixed with gate code. This tool is also extremely fast even when displaying very large schematics.

Application Notes

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Integration with Gate level netlist

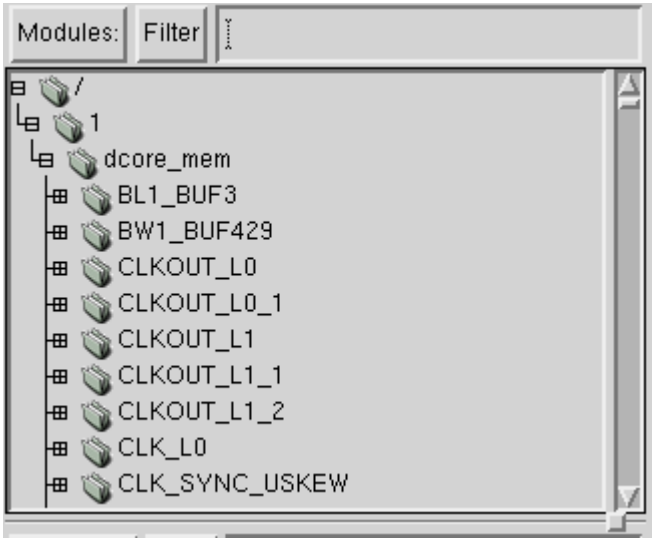
A new feature in Undertow allows designers to load in gate level design and debug the problems.

Couple of features have been implemented and more will be available in the future release.

To load in the gate level netlist, simply type in ut -iv <design>.v <library>.v

Undertow window should open, expand the hierarchy using the hierarchical brower on the left.

FIGURE 189. Hierarchical Browser



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To see the schematic, select the module and click on the right mouse button.

Note: Please make sure to select the module that you want to read. Simply loading in the top module and navigating to the desired module may take some times. As the gate level contains millions of gates, it will take a while to show the whole schematic.

To show logic across the boundry:

There are 2 ways to show the logic across boundry(hierarchy):

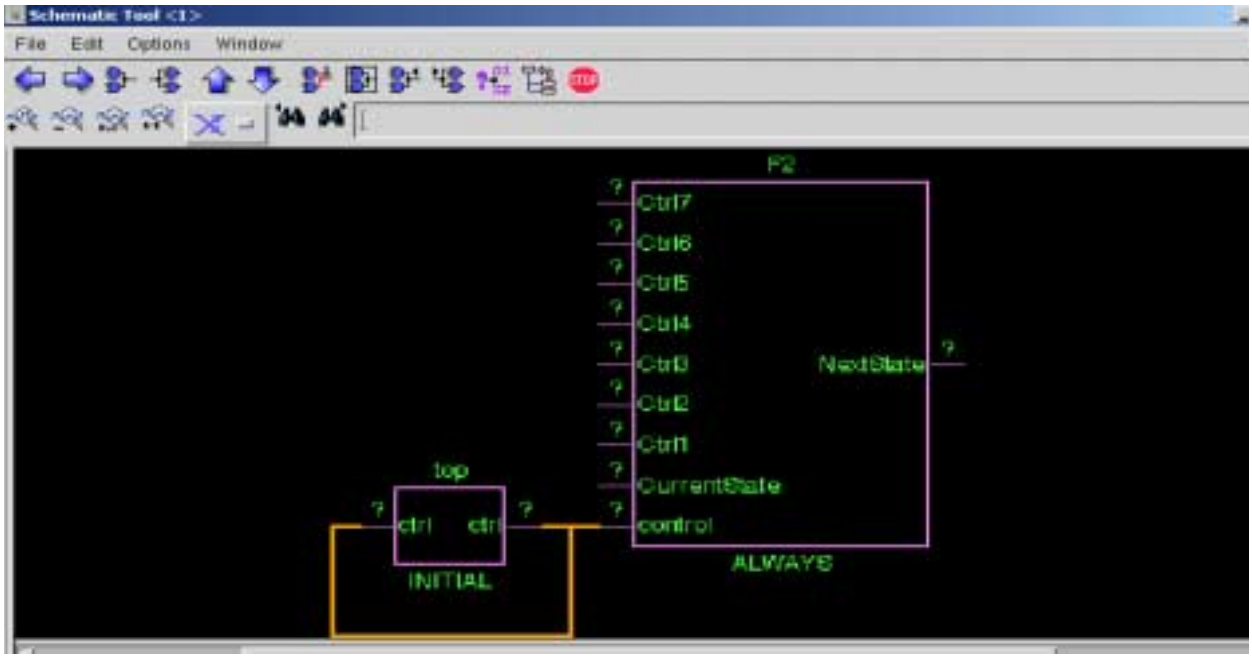
1. Selected the boundry pin, drag and drop the pin into a blank area

2. The schematic will show where this pin is connected to the other module

3. Select the pin on that module and click it.

4. Now the path across the module is shown

FIGURE 190. Schematic view of the wire going across module



Or

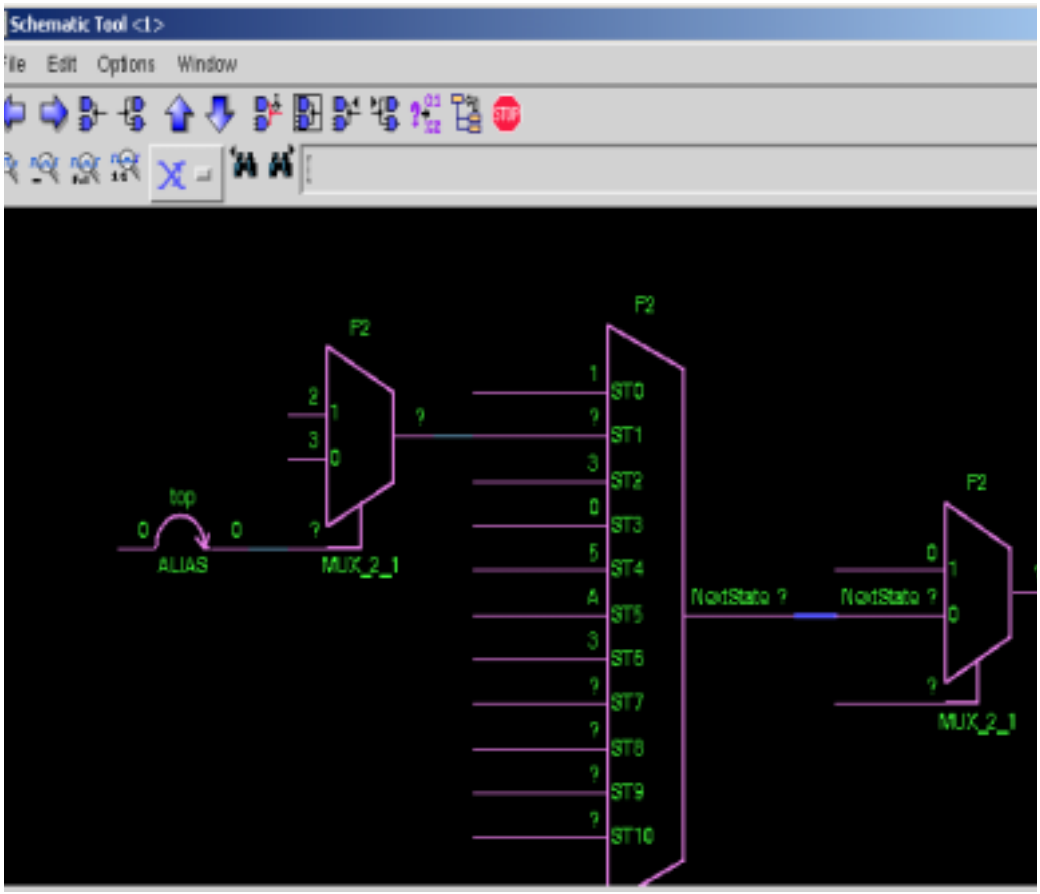
1. Select the boundry pin,

2. **Options->show maximum detail**s

3. Expand the driver forward

4. the path will be show despite the border as if boundry does not exist.

FIGURE 191. f Logic across module

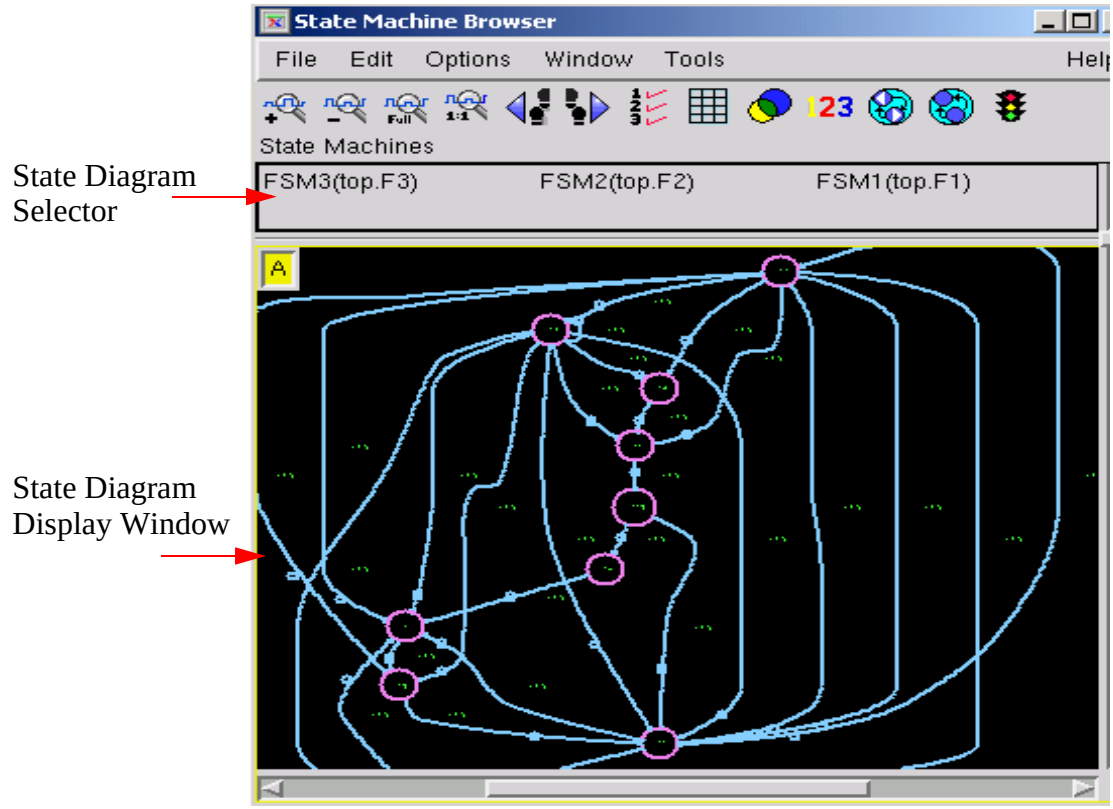


 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

State Diagram Window

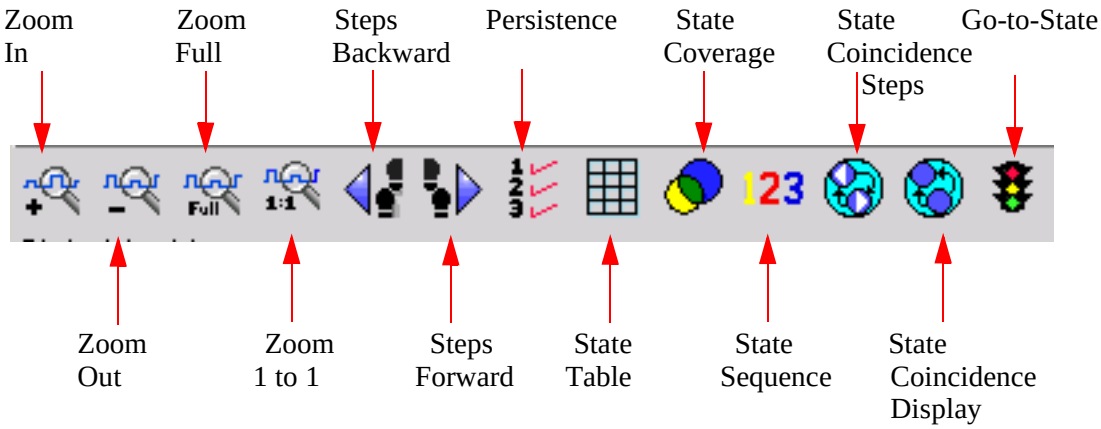
The State Diagram Window has 2 sections, the State Diagram Selector at the top of State Diagram Window and State Diagram Display Window. Selecting any state diagram in the selector will show that state diagram in the State Diagram Display Window (See Figure 192).

FIGURE 192. Selecting State Diagram for Viewing



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 193. State Machine Window toolbar



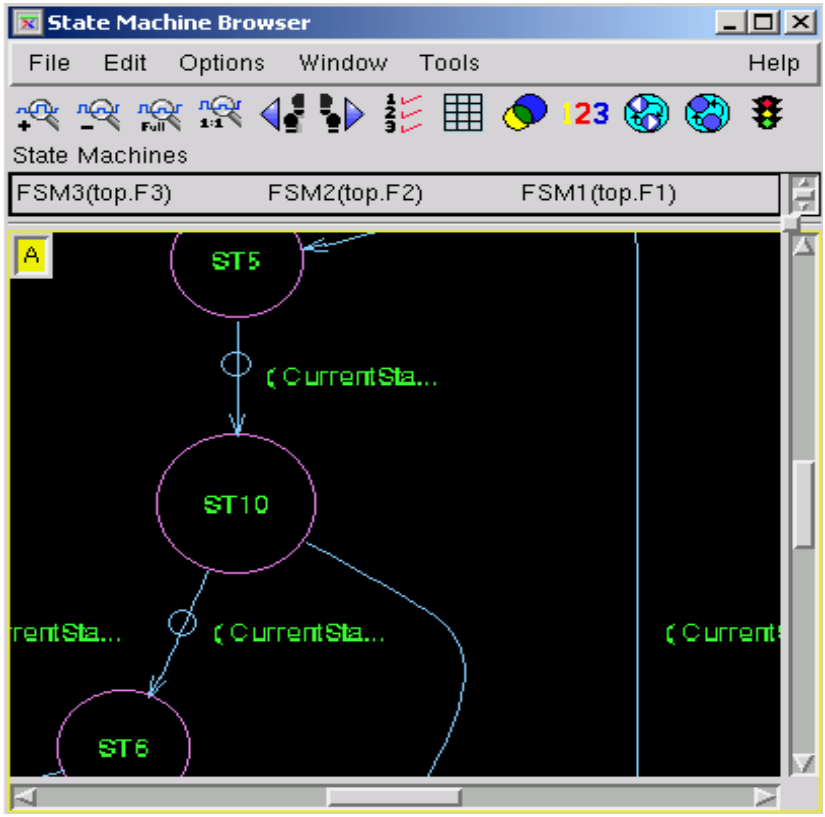
The State Diagram Window allows you to zoom into any area of the state diagram to see the state diagram detail better. You can step the state diagram forwards or backwards. The user can see as many state diagrams as they want at the same time. The State Diagram Window also includes a number of state diagram analysis tools.

To view a state diagram proceed as follows:

1. Select *FSM2*.
2. Box zoom to the lower 4 bubbles.
3. Step the state diagram forward and backward using the **Step Forward** and **Step Backward** icons. Note, the active state is shown with a double bubble and is always brought into view, if it was out of view or was on a display area and the stepping would have caused it to go out of view.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

FIGURE 194. Zoomed view in State Diagram Window



Note the state diagram is based on the source code, the active state, and the ability to step the state diagram is based on the values in the Waveform Window.

Also, note as you step the state diagram, the T0 cursor will be positioned to the exact time point in the waveform file where this state at this time becomes active, the T0 time is listed above the icons in the Waveform Window.

Viewing multiple state diagrams

In order to view more than one state diagram at the same time, do the following steps:

1. On the State Diagram Window press *Windows =>Add Pane Horizontal*. Then select another state diagram, *FSM1*. Note the active indicator is on (yellow) for the last pane brought up. If you select the active indicator it will now make that pane active.

   Note: The active indicator means that all icons apply to that pane and machine selector applies to that active pane.

2. Now press S**teps Forward** and **Steps Backward** icons for the *FSM1* that is now active.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Make the *FSM2* state diagram active by selecting its **Active** icon and then press the **Steps Forward** and **Steps Backward** icons to show that the state diagram that steps in sync with these icons is the state diagram that is active. All other state diagrams step in sync with the active state diagram.

State Machine Tools

The State Diagram Window has a number of tools to help designers analyze their state machine designs. These are as follows:

- **Persistence** - To record which states the **Steps Forward** and **Steps Backward** icons have caused the state diagram to transition through.
- **State Table** - To provide a quick view of the possible transitions between all of the "To" states from the "From" states.
- **Coincidence** - To illustrate for any state in a particular state machine, what are the states that have transitioned in the other state machines.

Persistence

1. Toggle down the **Persistence** icon. Note that all states in every state diagram displayed turns red in color.
2. Now step the active state diagram(use *FSM2*), forward several times using the **Steps Forward** icon. Note that several states have now turned from red to yellow.

State Table

1. Press the **State Table** icon. The State Table for the active state diagram, use *FSM2*, will be displayed.
2. Change the active state diagram from *FSM2* to *FSM1*, by pressing on the Active (A) icon in the pane holding the *FSM1* state diagram.
3. Now press the **State Table** icon again to see the State Table for FSM1.

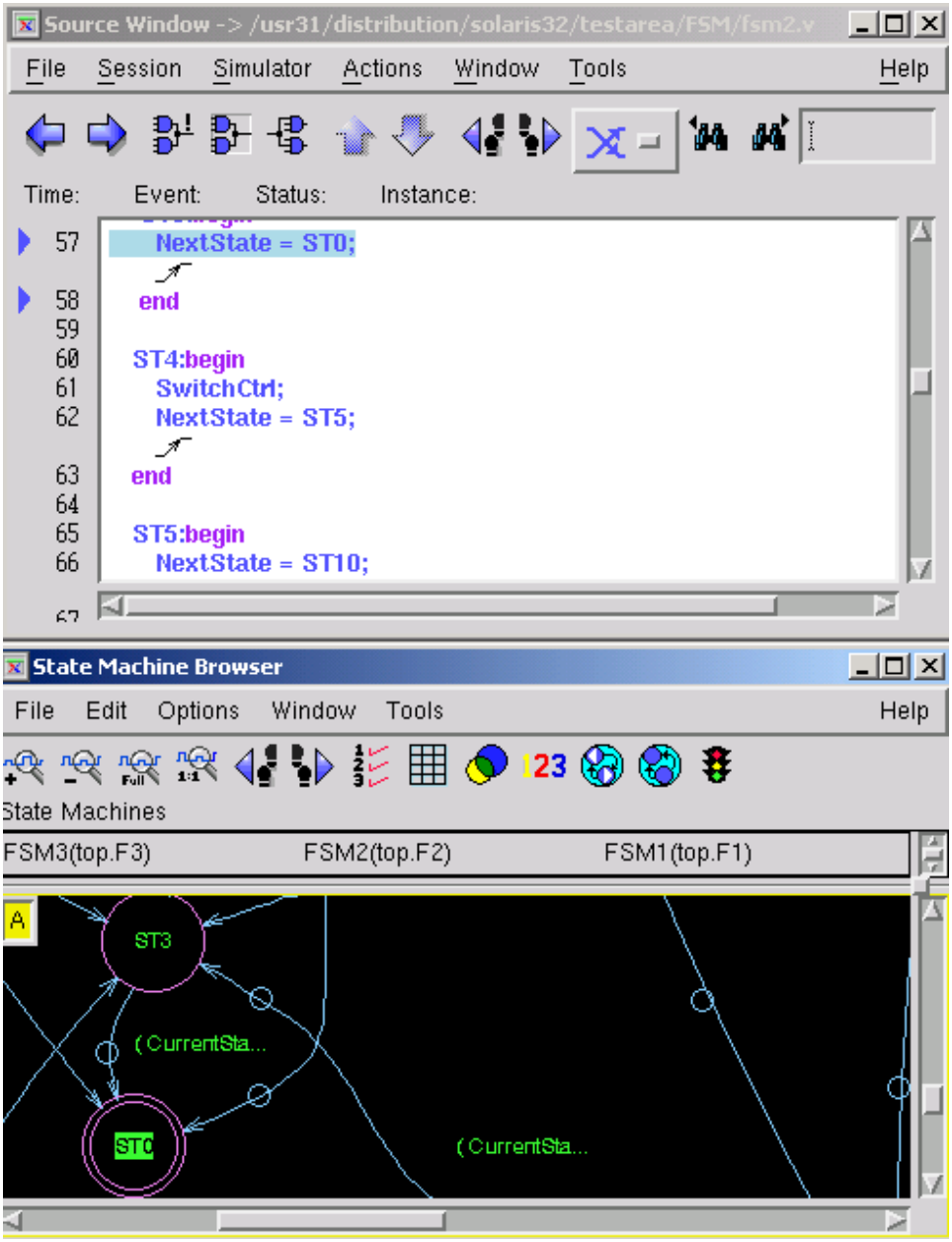
Overall Demonstration

1. First step the FSM2 state diagram to the ST0 state. Make sure *FSM2* is active.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. To find the ST0 state in the source code, drag the ST0 state from *FSM2* in the State Diagram Window to the Source Code Window. This will highlight the line of source code where the ST0 state has been assigned to the signal *NextState*, on line number 57, of file *fsm2.v* (See Figure 195).

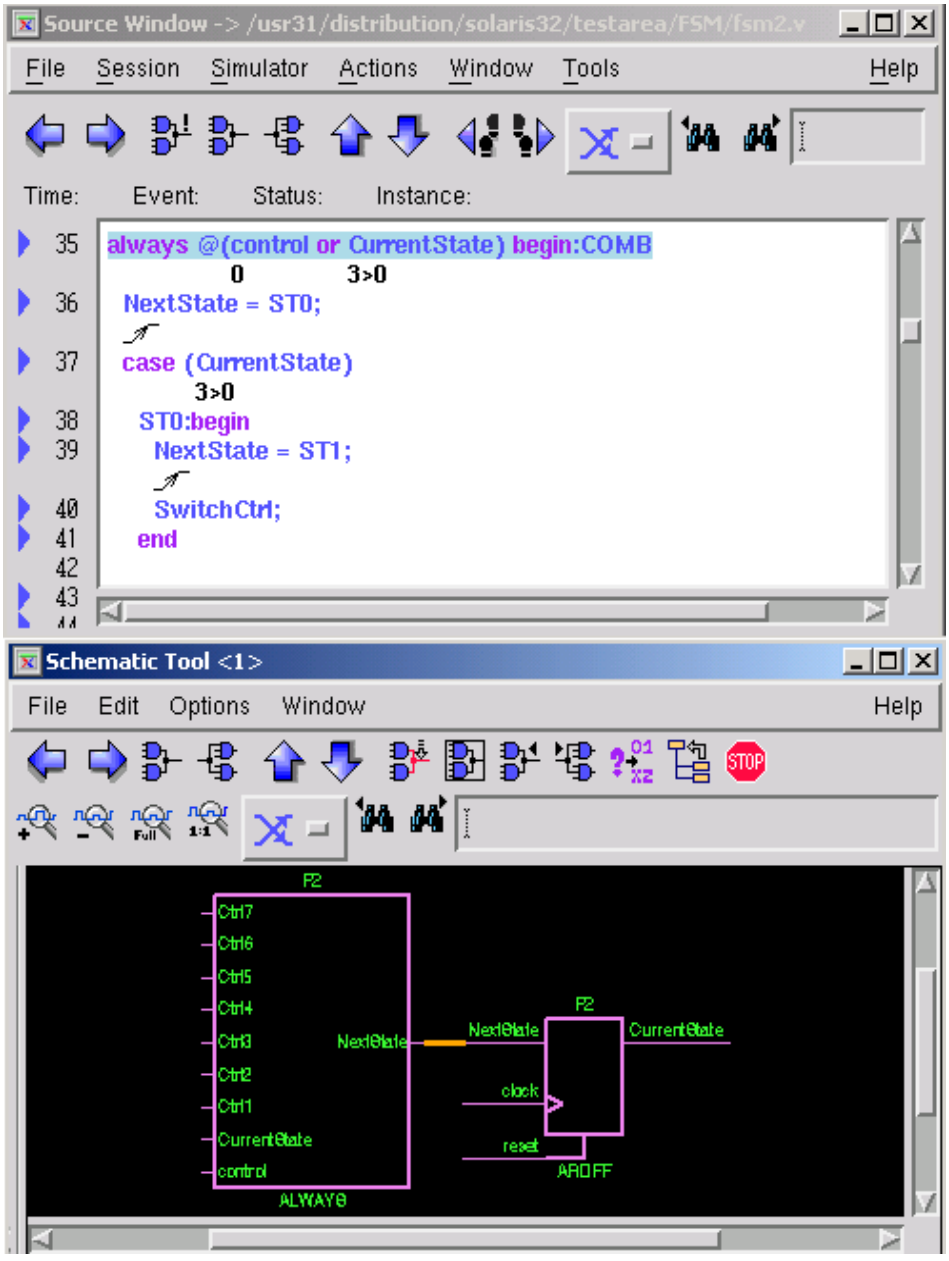
FIGURE 195. Dragging ST0 from the State Diagram Window to the Source Code Window



3. Now drag the signal on line 57, *NextState*, to the Schematic Window to find where this signal is to be found on the Schematic Window. It will be highlighted in an orange color.

   Notice that in the Schematic Window, “Next State” is a vector with 2 bits. This wire is going from an *always* block in the *top.f2* to a *FF* in *top.f2*. The output of the *FF* is labeled *CurrentState* (See Figure 196).

FIGURE 196. Dragging *NextState* from the Source Code Window to the Schematic Window



 Copyright Veritools 2004 All Rights Reserved

4. Next drag and drop *NextState* and *CurrentState* signals from the schematic to the Signal Name Area on the Waveform Window, at the location in the signal list where you want them to go.

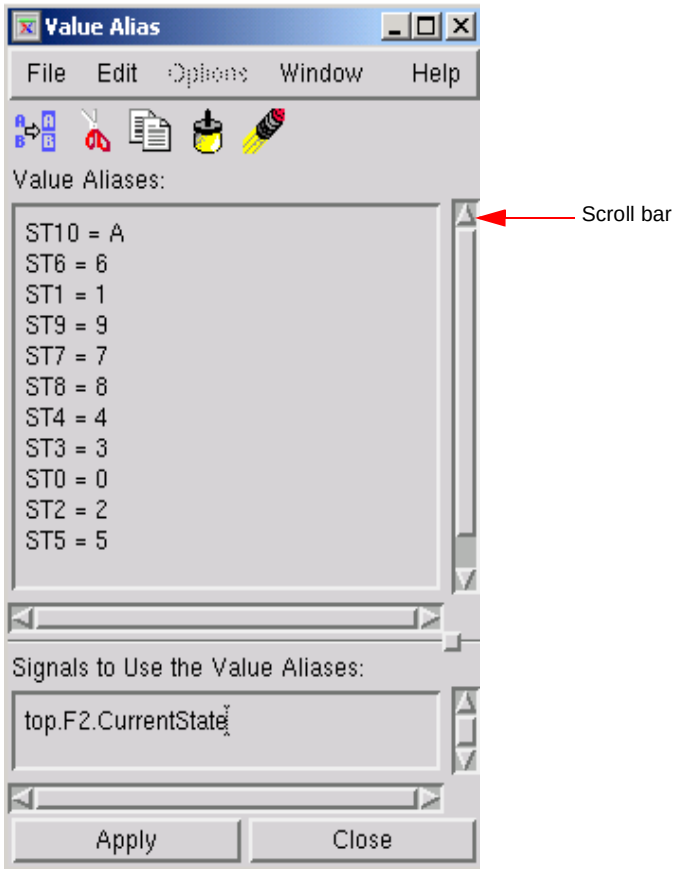
5. Notice the *CurrentState* signal is annotated to the name of the state, in this case ST0. See Signal Value Area to see this annotation.

   You can see that this annotation is due to an alias table by doing the following in the Waveform Window:

   Options =>Alias, Value=>List top.F2.CurrentState

   This shows the alias table for this signal, with the waveform annotation information(See Figure 197).

FIGURE 197. Alias Table provides information about Waveform Window annotations



6. Understanding the State Diagram

   A. Select the signal *CurrentState* on the Waveform Window.

   B. Press the **Go To** icon to bring up the Go To Window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

C. Press the **Go To** button and now note that the value you see in the Signal Value Area is ST1, the state Diagram has gone to ST1.

D. Do this again, and see that the value area is set to ST2, and that the state diagram has gone to the ST2 state.

E. Hence pressing the "GoTo" button on GoTo Window is identical to pressing the "Steps Forward" button on State Diagram Windows, when you have the current variable ("1/top/CurrentState[1:0]") selected in the Waveform Window.

Interactive Mode

The prior quick start illustrated using the Undertow Suite in a batch process. The files that were used could have been produced with any Verilog simulator that has trace capability (Verilog -XL, VCS). If you use a simulator without trace capability, the only feature the user will lose will be the virtual simulation capability.

Undertow Suite has facilities built in not only for use in a batch process, but also to drive any of the current Verilog simulators interactively. When running a simulation interactively, Undertow Suite can be used with trace enabled to allow virtual simulation after killing the simulator, or with trace disabled.

With No Trace Enabled

To the run_batch command line in the previous batch example, we used:

ut -iv -f fsm.vc -sigfile fsm.sigs -tracefile fsm.trace

But for the interactive mode we will add the type of simulator and name of simulator; in this case we will specify the simulator type to be Verilog-XL as follows:

ut -iv -xl verilog -f fsm.vc -sigfile fsm.sigs

Now run the above command line from the FSM directory.

Note: Verilog should have the Veritools PLI linked into it, and either be at the directory where your Verilog files are located, or have set in your .cshrc file the path to the directory holding the Verilog. For more information on linking PLI for this example see "Linking in the PLI and Building a Verilog Executable, Verilog-XL" on page 546.

 Copyright Veritools 2004 All Rights Reserved

The Undertow Suite will come up just as in batch mode, and if there is an existing waveform file signals can be selected and displayed.

Running Simulations

To choose the simulator and run the simulation do the following:

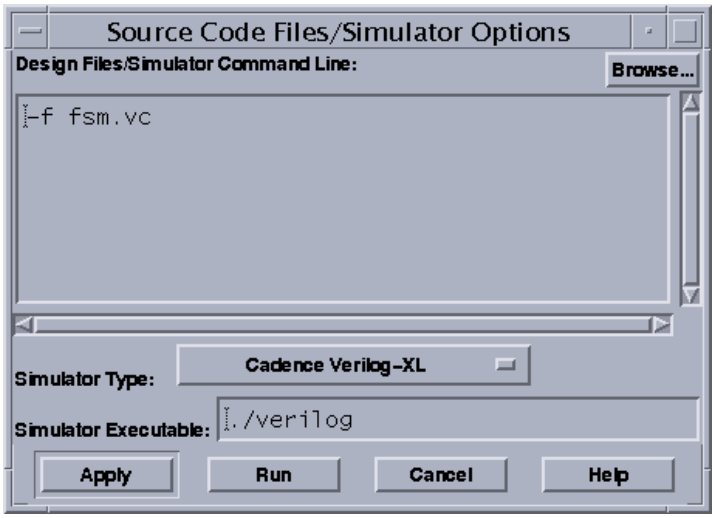
1. From the Source Code Window menu choose:

   Session => Source Code Files/Simulator Options

2. From the Source Code Files/Simulator Options Window, select the simulator type and enter the path in the Simulator Executable field.

3. Press **Apply** then **Run** (See Figure 198).

FIGURE 198. Source Code Files/Simulator Options



Another way to run the simulation is by doing the following:

1. Set the simulator options and have the location for the executable added to you computer system path.

2. Run the command line for the interactive mode listed in the first part of this section.

3. Run the simulation. From the Source Code Window choose:

   Simulator => Run

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Note that two new windows will now appear at the bottom of the Source Code Window, a Detail Window with the output text from the simulator, and a small Text Input Window below this for entering simulator commands manually.

Two icons will appear on the right of the Text Input Window, a **History** icon to reenter any previously entered command, and a **Buttons** icon. Pressing the **Buttons** icon will cause a list of simulator commands to drop down from the icon.

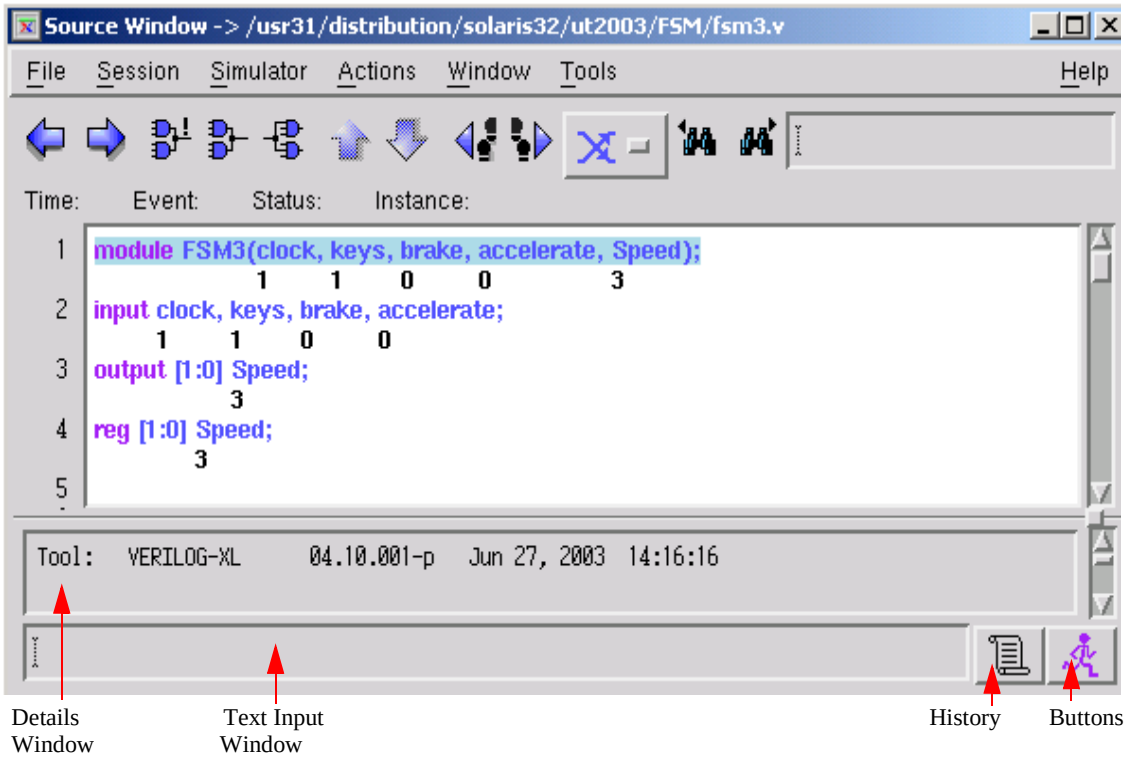
4. Press one of these buttons in the Buttons icon, for example **Go 1000 and stop**. The simulator will advance 1000 time units and then stop. The following text will appear in the Details Window:

   #1000 stop

   $stop at simulation time 1000

   You may have to scroll up in the Details window to see both lines of text, depending upon the size of your window (See Figure 199).

FIGURE 199. Source Code Window with interactive mode set



5. Now press the **Step Forward** icon. This will advance the simulator one event per press of the icon.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

*Note: When trace is disabled the user cannot use the **Step Backward** icon to step the simulation backward. However, if you use **Step Forward**, then you can **Step Backward** through those steps.*

6. To view, add, or modify the buttons controlling the interactive simulation, do the following on the Source Code Window:

   Sessions => Button

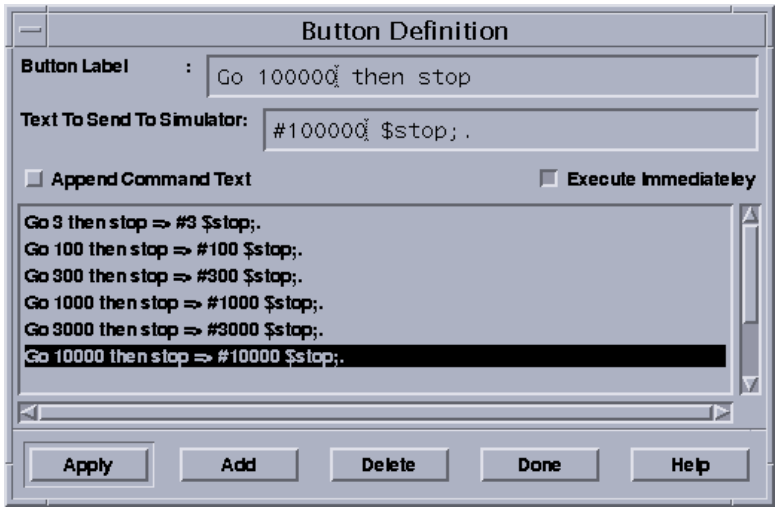
   This window will display the list of buttons and the command that is to be sent to the simulator for each button.

7. Click on the **Go 10000 Then Stop => # 10000 $stop** from the list in the Button Definition Window.

8. In the **Button Label** field add a zero to the name, and in the **Text To Send To Simulation** field also add a "0".

9. Press **Add** and then **Done** to add this new button (See Figure 200).

FIGURE 200. Button Definition Window displaying *Go 100000 then stop*



10. Go to the Buttons icon and select the newly created choice.

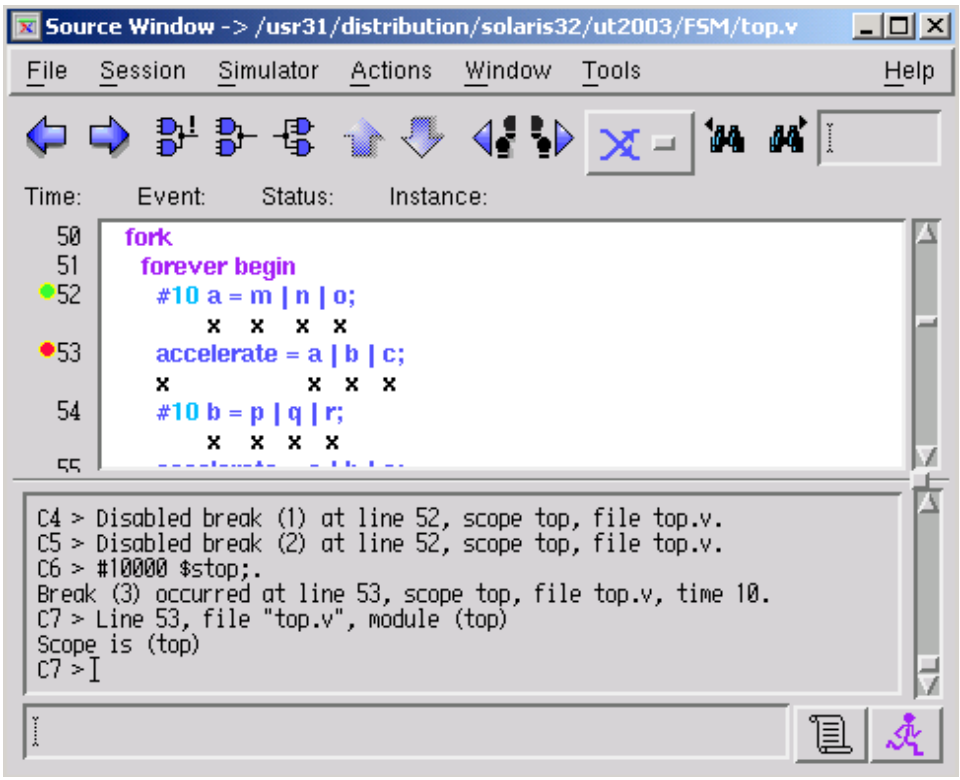
Breakpoints

1. To add break points, select the area to the immediate left of any line number. Note some lines will not accept any break points.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Pressing *Simulator => Continue*, starts the simulator and run until it hits a break point (See Figure 201).

FIGURE 201. Setting breakpoints


Source Window -> /usr31/distribution/solaris32/ut2003/FSM/top.v
File Session Simulator Actions Window Tools Help
Time: Event: Status: Instance:
50 fork
51 forever begin
52 #10 a = m | n | o;
53 accelerate = a | b | c;
54 #10 b = p | q | r;
C4 > Disabled break (1) at line 52, scope top, file top.v.
C5 > Disabled break (2) at line 52, scope top, file top.v.
C6 > #10000 $stop;.
Break (3) occurred at line 53, scope top, file top.v, time 10.
C7 > Line 53, file "top.v", module (top)
Scope is (top)
C7 >

3. Select any set break point to turn the break point from red to green. A green breakpoint does not stop the simulator.

4. Pressing the Shift key down while selecting the breakpoint turns the breakpoint yellow. This is to stop at the breakpoint and then turn the color of the breakpoint to green.

5. Pressing the Control key while selecting the breakpoint removes the breakpoint.

Trace Enabled

To rerun the simulation with trace enabled proceed as follows:

1. To exit the simulator, do the following in the Source Code Window:

   Simulator => Kill

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

A pop-up window states that the Simulator has exited. Select the **Dismiss** button to close the pop-up window.

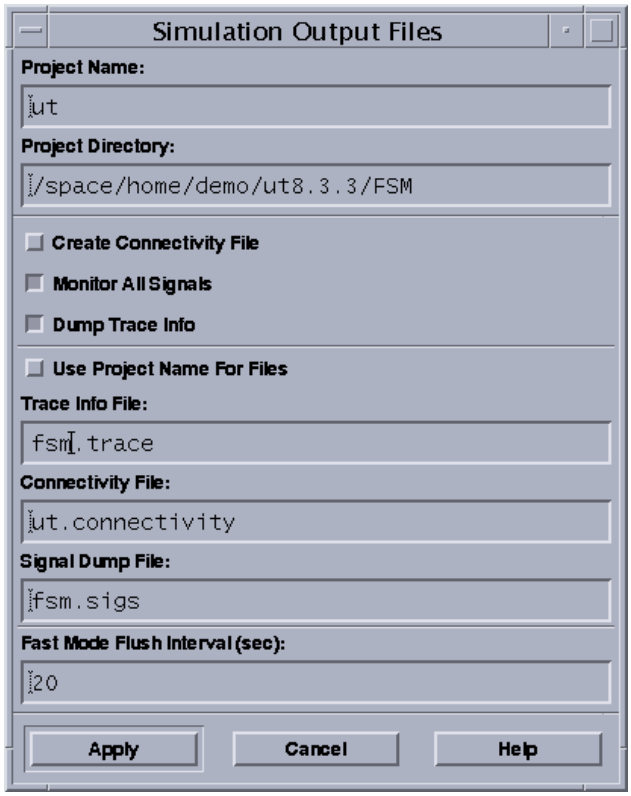
2. Bring up the Simulation Output Files in the Source Code Window:

   Sessions=>Simulation Output Files

3. Select the option **Dump Trace Info** to turn trace on, and enter the trace file name *fsm.trace* in the **Trace Info File** field.

4. Click **Apply** and then **Cancel** button (See Figure 202).

FIGURE 202. Simulator Output Files Window to set tracing on



5. Start the simulation::

   Simulator => Run

The simulator will run as before. However, a tracefile will now be written out and loaded into the Undertow Suite.

This trace file will allow the user to:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

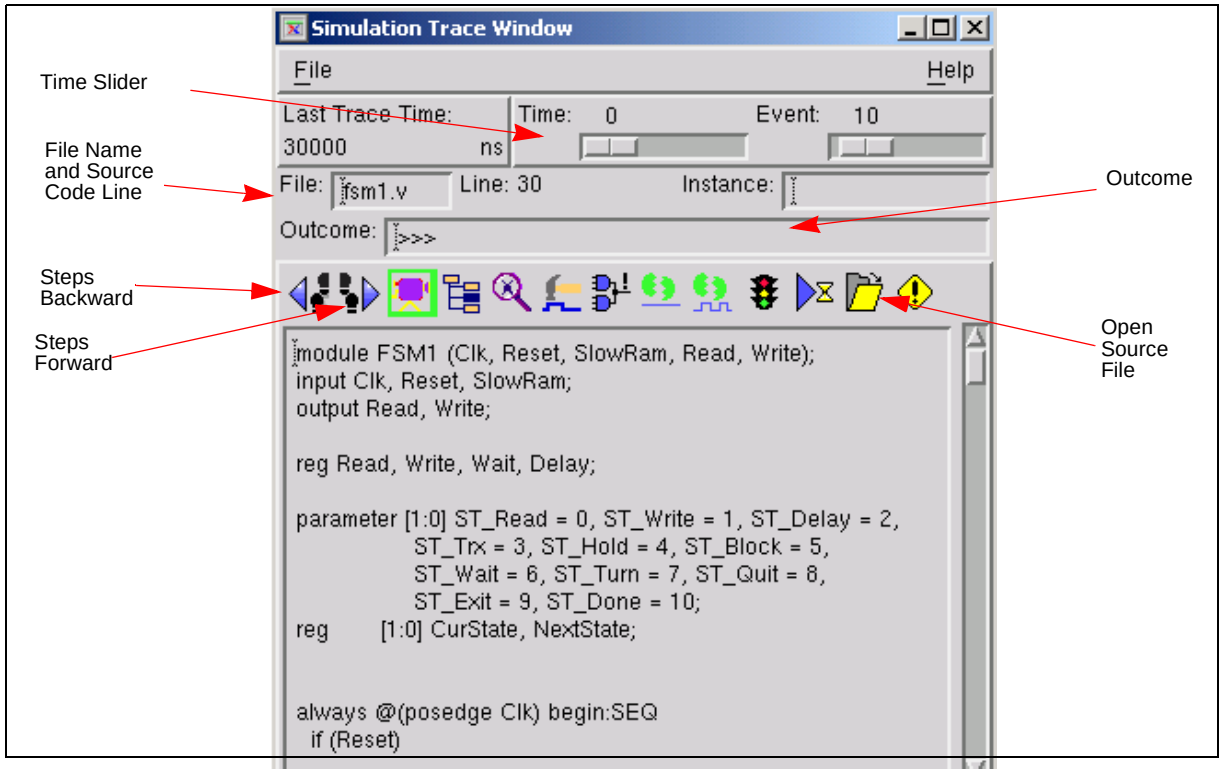
1. Step the simulator backward to review sequence of simulation by pressing on the Source Code Window **Steps Backward** icon.

2. Bring up the Trace Window:

   Tools => Trace

   When the Trace Window is up, the user can step this trace file forward or backward by pressing **Steps Forward** or **Steps Backward** icons. When the user presses Steps Forward and the simulation is at the end of the current simulation point, the simulator will be commanded to step one more event for each press of the Steps Forward icon (See Figure 203).

FIGURE 203. Trace Window stepping through events



The simulator can be killed at any time, and the user can still use the trace file and Undertow Suite as a virtual simulator to see the exact sequence of the simulation at any point in the simulation run.

To run the interactive mode without trace in an earlier example we had used the command line:

ut -iv -xl verilog -f fsm.vc -sigfile fsm.sigs

 Copyright Veritools 2004 All Rights Reserved

For the interactive mode with trace in the last example we could have started by using the command line:

ut -iv -xl verilog -f fsm.vc -sigfile fsm.sigs -tracefile fsm.trace

This is the end of the interactive mode section and the Verilog Quick Start chapter. To create the file types used in this example from your own data refer to the following sections (See "Creating A Trace Dump File, Using Cadence Verilog-XL" on page 548.) (See "Linking in the PLI and Building a Verilog Executable, Verilog-XL" on page 546.).

Linking PLI Into Your Verilog

You have to link in the PLI into your Verilog in order to make a batch file for use with Undertow Suite in the batch mode or to use Undertow Suite in interactive mode.

Linking in the PLI and Building a Verilog Executable, Verilog-XL

- Set the **UT_ROOT_DIR** environment variable, to the Veritools/Undertow *install* directory.
- Set the **CDS_ROOT** environment variable, to the Cadence *install* directory.

Veritools supplies two files to build into the **verilog** executable. They all reside in the **$UT_ROOT_DIR/PLI** directory, and are called **vtpli.o** and **vt_veriuser.c**

1. Run the **vconfig** utility, supplied with your Verilog distribution.

   This utility produces a script, which creates a new Verilog executable.

   *Note: The **vconfig** program prompts you for various information, and indicates the default values.*

2. Use the defaults, which are usually the right choices.
3. When asked how to link PLI, choose **PLI static**.

   vconfig displays the following prompt:

   The user template file **'veriuser.c'** must always be included in the link statement. **What is the path name of this file?[/cadence/tools/verilog/src/veriuser.c]:**

4. To respond to this prompt, type the complete path name to the **vt_veriuser.c** file.

   vconfig displays the following prompt:

   > **Please list any other user files to be linked with this Verilog-XL. Files ending with '.c' will be taken as C source files and compiled before being linked with Verilog-XL. All other files will be added to the final link command as-is. For C++ linking in the Standalone mode, you must manually edit the resultant shell script. List the files one at a time, terminating the list with a single '.'**
   >
   > ---------->

5. Respond with the complete path name to the **vtpli.o** file.

For Making Batch Files Only, Adding in System Tasks

The commands $vtDumpvars and $vtDumpnames are two system commands that are only used when you make batch files running Undertow Suite in batch mode. In addition you have to use a +VTCOMPRESS command to specify the amount of compression (See "Command Line Options" on page 548.)

If you want to run Undertow Suite in interactive mode, these two commands are sent automatically to the simulator.

The following is an example of a code modification made to the *top.v* file. Search for *$vtDumpvars* in the file and modify.

```
initial begin
  $vtDumpnames("fsm.sigs");
  $vtDumpvars();   /*dumps everything*/
 // $utConnectivity;
 #30000 $finish;
end
```

- **$vtDumpvars** creates a **.sigs** file, and uses the same arguments as *$dumpvars*. It can replace *$dumpvars* in your testbench. $vtDumpnames system task can only be started once, and must precede the $vtDumpvars system task.
- **$vtDumpnames** changes the name from a default of *vt.sigs* to a name you choose, in this example *fsm.sigs*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Command Line Options

These are the command line options that the user sends to the Verilog simulator.

- **VTCOMPRESS200**—Real time mode. This Fast File is quick to create, easy to read, and produces a small file size which can be read as the simulator is running.
- **VTCOMPRESS250—**Like VTCOMPRESS200, but with very small file and higher compression ratio.

Creating A Trace Dump File, Using Cadence Verilog-XL

The trace dump file records each line of source executed as the simulator is running in a trace file. It is only required if you are using Undertow Suite to do source-code-level debugging, and you wish to use the virtual simulation feature. Virtual simulation is the ability to run Undertow Suite without checking out a simulator license, while having all of the apparent capability of running an active simulation.

If you use the Cadence Verilog-XL simulator, you can start the **vtrun** executable to create Trace dump files. Since the trace operation is built into Verilog-XL, the Veritools executable **vtrun** is used to pass to Verilog-XL the command **-trace**; and then to take the resulting trace information and reformat and compress it for use by Undertow Suite.

- When you start the **vtrun** executable, use all of the parameters that you would normally use to invoke the Verilog simulator.
- Additional parameters, specific to **vtrun**, control how you generate a **trace dump** file:
  - The **vtrun -trace** command line argument generates a trace dump file, using the default file name, vt.trace.
  - The **vtrun -tracefile *traceFileName*** command line argument specifies the name of the trace dump file, where *traceFileName* is the name that you want to assign to the trace dump file. For example to run the example as previously described in the batch example the complete command line is the following:

    vtrun -xl verilog -f fsm.vc -trace -tracefile “fsm.trace”

Running Simulation

1. If you use **$vtDump**, you do not need to select any options when you run simulation.
2. If you use **$vtDumpvars**, you can select options such as:

   % my_vlog +VTCOMPRESS200

Viewing Results

- **ut -v <file_name>.fast** loads a **$vtDump** file directly into Undertow.
- **ut -iv -sigfile <file_name>.sigs -f vfiles** loads a netlist, and the specified **$vtDumpvars** file, into the interactive tool.

  For detailed operation of Veritools PLI routines, refer to .Chapter 9, “Working With PLI”

Using Other PLI Routines

Installing Veritools PLI Routines for VCS

Adding the Veritools PLI to VCS (on Solaris, HP-UX, Linux platforms)

- Set the **UT_ROOT_DIR** environment variable to the Veritools/Undertow *install* directory.
- Set the **VCSI_HOME** environment variable to the VCS *install* directory

STEPS for SIMULATING USING VCS:

Adding PLI System Tasks to the Testbench

Using **$vtDump** creates a **.fast** file. Place this file into the testbench, before the standard **$dumpvars** Verilog task**.**

- **$vtDump** pipes the *vcd* output through the **$UT_ROOT_DIR/vdump** utility.
- You must include the **$UT_ROOT_DIR** in your **$path** environment variable.
- **$vtDumpvars** creates a **.sigs** file, and uses the same arguments as *$dumpvars*. It can replace *$dumpvars* in your testbench.

From the simulation command line, you can use **$vtDumpvars** to choose better compression of a dumpfile, or less compression and faster simulation. The following are examples:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **VTCOMPRESS200**—Real time mode. This Fast File is identical to *vdump*, is quick to create, easy to read, and produces a small file size.
- **VTCOMPRESS250**—Like VTCOMPRESS200, but with very small file and higher compression ratio..

Building simv

Veritools supplies three files to build into the **simv** executable. They all reside in the **$UT_ROOT_DIR/PLI directory,** and are called:

- **iv.v**
- **vtplivcs.o**
- **vt_vcs.tab**

You can merge **vt_vcs.tab** with your **.tab** file.

Include these files after the **-P** option, on the **vcsi** command line. Buildg **simv** can be performed either in Batch mode or Interactive mode.The syntax for Batch mode is,for example

vcsi -Mupdate +vpi +rad +vcsd -f source $UT_ROOT_DIR/PLI/vt_vcs.tab -P $UT_ROOT_DIR/PLI/vtplivcs.o

vcsi -Mupdate +vpi +acc+2 -lm -line -f source -P $UT_ROOT_DIR/PLI/vt_vcs.tab $UT_ROOT_DIR/PLI/iv.v $UT_ROOT_DIR/PLI/vtplivcs.o -f vfiles

Invoking **vcsi** produces a **simv** executable.

*Note:You must include the **-line** option in your VCS compilation command, if you want to run Undertow Suite in an **interactive mode**. If you want to run VCS in batch mode, you do not need to put **-line** into the command line.*

For detailed operation of Veritools PLI routines, refer to .Chapter 9, “Working With PLI”

Running Simulation

1. If you use **$vtDump**, you do not need to select any options when you run simulation.
2. If you use **$vtDumpvars**, you can select options such as:

% simv +VTCOMPRESS200

Invoking Undertow and Viewing Results

- **ut -v <file_name>.fast** loads a **$vtDump** file directly into Undertow Suite.
- **ut -iv -sigfile <file_name>.sigs -f <vferilogfile nmes>** loads a netlist, and the specified **$vtDumpvars** file, into the interactive tool.

Sample Script For Running VCS

STEPS for SIMULATING USING VCS:

1)Source environment variables

For example,

envsource has all the environment variables set up. You can change the paths accordingly.

----------*envsource file contents*----------

setenv VCSI_HOME <path where VCSI is installed>

setenv DEFAULT_VCS_HOME <path where VCSI is installed>

setenv VCS_HOME <path where VCSI is installed>

setenv TMPDIR /tmp

setenv VCS_NETHOST vt1

setenv VCS_LTD_LICENSE 1

setenv VCS_CC cc

---

2)Add to source code close to top module

initial

begin

$vtDumpvars();

end

In this example, the above has been inserted in 'top.v' file.

3) To run simulator in batch mode, do as follows:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

./run_vcsi_batch

run_vcsi_batch : script for running all vcs commands.

---------*run_vcsi_batch contents*----------

#!/bin/csh -f

*vcsi -Mupda*te *+rad +vcsd -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab*

./simv +VTCOMPRESS250 +VTVECTORVALUES
---

a) -f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v iv.v is available in the example directory.

b) vtplivcs.o and vt_vcs.tab: are available in our PLI directory. You need to link in the Veritools PLI routines when you compile your model with VCS. The Veritools PLI routines use these files to generate a VCS compiled model. vt_vcs.tab defines all Veritools user-defined PLI definitons.

c) -P: used to specify the PLI tab file.

> *NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.*

The simulator runs and creates a vt.dump file.

Compilation:

vcsi <flags> -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab

To enhance Simulation speed, users can run VCS wth different options to enhance/hinder performance. Try avoiding the slow option and use the fast options. Here are the options users can use in VCS

Options for for increased performance in VCS compilation:

+rad: Perform aggressive optimization for RTL or (non timing) gate level

+vcsd: Use more efficient Direct Kernel Interface for VirSim dumping

-**Mupdate:** Use incremental compile to only recompile modules that

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

functionally changed

+rad+1: Perform localized optimizations only

+nospecify: Ignore path delays and timing checks for functional gat simulations

+notimingchecks: Ignore timing checks +timopt: Perform agressive timing optimizations (for SDF gate simulations)

+nocelldefinepli+2: Do not dump or use PLI access from within library cells

+2state: Convert entire design (except certain constructs) to 2 states for more speed/capacity.

Flags that hinder performance:

+cli: Turn on interactive debug for the entire design

+cli+mod=#: Turn on interactive debug just for module mod

+acc+2: Enables PLI ACC capabilities, including breakpoints

-line: Allow line stepping in debugger

-00: Turn off optimizations

-I: Obsolete flag for interactive GUI debug. Instructs VCS to automatically include the +cli (command line interface), -P virsims.tab (default VirSim PLI table) and -lm (math library) options. This option is required to create VCD+ files like the -RI option. The virsims.tab file defines the VCD+ system tasks such as $vcdpluson. Unlike the -RI option, simulation with VirSim does not start after compilation but allows post-processing of the VCD+ file after simulation

-RI: Compile and run interactive debug with VirSim GUI

-PP: Allow VirSim vcd+ binary dumping for post-processing debug. Does the same thing as the -I option but minimizes detail of net data for faster post-processing

+race: Turn on race detection

+prof: Turn on VCS profiling

-gen_c: Force generation of C intermediate code instead of native object code -P pli.tab which contains acc=rw, cbk:*: Globally turn on PLI access +multisource_int_delays: Enables global PLI access visibility

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

+transport_int_delay: Enables global PLI access visibilty-X*: version specific flags t o work around specific bugs

+vpi- Verilog PLI (VPI) library is enabled through this option

Recommended Use(for optimal speed):

vcsi -Mupdate +rad +vcsd -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab

./simv +VTCOMPRESS250 +VTVECTORVALUES

4) To use interactive mode:

./run_vcsi_int

-------run_vcsi_int contents----------

#!/bin/csh -f

vcsi -Mupdate +cli +acc -line -lm -f source $UT_ROOT_DIR/PLI/vtplivcs.o -P $UT_ROOT_DIR/PLI/vt_vcs.tab

---

The -line option must be given if you want to run Undertow Suite in interactive mode. The -line option is not required for running VCS in batch mode.

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

BATCH MODE:

ut -iv -f <source> -sigfile <dump filename>

NOTE: <source> is the file that contains the source code files for simulation.

For example,

ut -iv -f source -sigfile vt.dump

INTERACTIVE MODE:

ut -iv -vcs <vcs_simulator_executable> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code file names>"

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

For example,

ut -iv -vcs simv -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

for information about using -tracefile please refer to “Batch Simulation with Command Line” on page 288

Installing Veritools PLI Routines for Modeltech

Adding the Veritools PLI to MTI (on Solaris, HP-UX, Linux platforms)

For detailed operation of the Veritools PLI routines, refer toChapter 9, “Working With PLI”

Adding PLI System Tasks to Testbench

Using **$vtDump** creates a **.fast** file. Place this file into the testbench, before the standard **$dumpvars** Verilog task**.**

- **$vtDump** pipes the *vcd* output through the **$UT_ROOT_DIR/vdump** utility.
- You must include the **$UT_ROOT_DIR** in your **$path** environment variable.
- **$vtDumpvars** creates a **.sigs** file, and uses the same arguments as *$dumpvars*. It can replace *$dumpvars* in your testbench.

From the simulation command line, you can use **$vtDumpvars** to choose better compression of a dumpfile, or less compression and faster simulation. The following are examples:

- **VTCOMPRESS200**—Real time mode. This Fast File is identical to *vdump*, is quick to create, easy to read, and produces a small file size.
- **VTCOMPRESS250**— Like VTCOMPRESS200, but with very small file and higher compression ratio.

Compiling and Simulating with MTI

1. After you set the **PLIOBJS** environment variable, compile as you normally would.

   For example:

   vlog -f vfiles

2. If you use **$vtDump**, you do not need to select any options when you run simulation.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. If you use **$vtDumpvars**, you can select options such as:

   vsim -c top +VTCOMPRESS200

You can also run the simulator from the Undertow Interactive tool:

ut -iv -modeltech vsim top -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

1. To open the interactive tool, type:

   ut -iv

2. Select the ***Sessions*** **=>** ***Source Code Files/Simulation Files*** menu item.
3. Select the type of simulator to use.
4. Specify the path to the simulator executable.
5. Add simulation files.
6. Select **browse**.
7. Select either:
   - Individual files to load, or
   - A list of files that contain valid Verilog options.
8. Click apply.
9. The file loads into the **design files** window.
10. If you use a single file that lists files and options, add a **-f** and a space, in front of the path to the file.
11. Add simulator-specific options to the simulator command line.
12. For MTI, an option might be something like:

    -c top

13. Click the Apply button.

    The netlist loads into the **Source** Window.

14. Select ***Simulator*** **=>*Run*** to start the simulation.

Adding the Veritools FLI to MTI 5.3+ (Solaris, HP-UX,

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Linux)

Set your environment variable and paths:

1. Set the **UT_ROOT_DIR** environment variable, to the Veritools/Undertow *install* directory.
2. Set the **ModelTech** environment variable, to the **bin** directory of the ModelTech *install* directory.
3. Set the **PLIOBJS** environment variable to *$UT_ROOT_DIR/FLI/fli_modtech_53.so*

Adding FLI System Tasks to the Testbench

Create a foreign architecture. The following is an example file, called **foreign.vhdl**

```
entity dump_design is end;
architecture only of dump_design is attribute foreign: string;
attribute foreign of only: architecture is "dump_design_init
  $UT_ROOT_DIR/FLI/fli_modtech_53.so";
begin
End;
```

Compiling and Simulating with MTI

1. After you set the **PLIOBJS** environment variable, compile as you normally would.

   For example:

   vcom foreign.vhdl <rest of design files if needed>

   Run **vsim** as you normally would.

   This creates a **vhdl.dump** file, based on the **VTCOMPRESS200** mode.

Viewing Results

- **ut -v vhdl.dump** loads a file directly into Undertow Suite.
- **ut -iv -sigfile vhdl.dump -f vfiles** loads the netlist and waveform file into the interactive tool.

For detailed operation of Veritools PLI routines, refer to Chapter 9, “Working With PLI”.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Script file for running MODELTECH

Sourcing Env. Variable, For example,

./envsource

----------envsource file contents----------

setenv PLIOBJS $UT_ROOT_DIR/PLI/vtpli_modtech.so

#setenv ModelTech <path where ModelTech has been installed>

---

vtpli_modtech.so is available in our PLI directory.

Add to source code close to top module

initial

begin

$vtDumpvars;

end

In this example, the above has been inserted in 'top.v' file.

Run simulator as follows:

./run_modeltech

run_modeltech : script for running all modelsim commands.

---------*run_modeltech contents*----------

#!/bin/csh -f

if (-e work) then

rm -r -f work

endif

if (! -e work) then

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

vlib work #creates new design library work

endif

vlog -f source #compiles the verilog files into the work library

vsim -c -do 'run -all' top vtInteractive +VTCOMPRESS250 +VTVECTORVALUES

#vsim -c -do 'run -all' top vtInteractive

---

a) **-f <file> :** used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

b) **-c :** run in command line mode

c) -**do** '**run -all**': to run until there are no more simulation events or until $finish is executed in the Verilog code.

d) **top and vtInteractive:** are the top level modules in this example. The simulator vsim is invoked with the top level modules.

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.

Viewing Model Tech Waveform

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

<source_code_file> is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

 Copyright Veritools 2004 All Rights Reserved

For example,

ut -v vt.dump

INTERACTIVE MODE

ut -iv -modeltech <simulator_executable_name> <top level module names> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code files>"

For example,

ut -iv -modeltech vsim top -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Installing Veritools PLI Routines for NC Verilog

Set your environment variable and paths

1. Set the **UT_ROOT_DIR** environment variable, to the Veritools/Undertow *install* directory.
2. Set the **CDS_INST_DIR** environment variable, to the Cadence *install* directory.

Modifying the cr_inca Script

Veritools provides this script, to use in place of the **Makefile.nc** file that Cadence provides.

1. Download the appropriate **cr_inca** script from the www.veritools.com web site.
2. Change the following environment variables, to point to your own file system locations:

- **CDS_INST_DIR**—the path to your Cadence *install* directory.
- **PLI_BUILD**—set to **$UT_ROOT_DIR/PLI**

f the vt.dump file.Optimization is more in the case of designs with large number of vectors.

Adding PLI System Tasks to Testbench

Using **$vtDump** creates a **.fast** file. Place this file into the testbench, before the standard **$dumpvars** Verilog task**.**

- **$vtDump** pipes the *vcd* output through the **$UT_ROOT_DIR/vdump** utility.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- You must include the **$UT_ROOT_DIR** in your **$path** environment variable.
- **$vtDumpvars** creates a **.sigs** file, and uses the same arguments as *$dumpvars*. It can replace *$dumpvars* in your testbench.

From the simulation command line, you can use **$vtDumpvars** to choose better compression of a dumpfile, or less compression and faster simulation. The following are examples:

- **VTCOMPRESS200**—Real time mode. This Fast File is identical to *vdump*, is quick to create, easy to read, and produces a small file size.
- **VTCOMPRESS250**— Like VTCOMPRESS200, but with very small file and higher compression ratio.

Creating the NC executables

Run **cr_inca**. This creates the **ncelab** and **ncsim** executables.

Running ncprep, ncvlog, ncelab, ncsim

1. Execute the following commands:

   ncprep -f vfiles +overwrite

   /* prepares database and creates .args for ncvlog, ncelab and ncsim */

   ncvlog -f ncvlog.args

   /*compiles the design */

2. Edit the **ncelab.args** file, and add the following to it:

   -ACCESS +RCW

3. Run the following:

   ncelab -f ncelab.args

   /*this elaborates the design */

4. Edit the **ncsim.args** file, and add the following lines:

   -PLINOWARN

   -TCL

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

5. Run the following:

ncsim -f ncsim.args

/* this runs the simulation */

Viewing Results

- **ut -v <file_name>.fast** loads a **$vtDump** file directly into Undertow.
- **ut -iv -sigfile <file_name>.sigs -f vfiles** loads a netlist and the specified **$vtDumpvars** file into the interactive tool.
- For Interactive Simulation: ut -iv -ncverilog ncsim -sigfiles ut.sigs -ivsimcmp "-f source"

For detailed operation of Veritools PLI routines, refer toChapter 9, "Working With PLI"

Sample Script for running NCSIM:

1)Source environment variables

You can setup your environmental variables as follows

./envsource

envsource has all the environment variables set up. You can change the paths accordingly. Make sure that the $CDS_INST_DIR is defined for your Cadence installation directory.

vt_veriuser.c and vtplinc.o are available in our PLI directory.

----------*envsource file contents*----------

#setenv CDS_INST_DIR <cadence installation directory>

setenv CDS_INST_DIR /cad_tools/LDV5.1

setenv ittsimUndertowSeDir $CDS_INST_DIR/tools/dfII/local/undertow

setenv LD_LIBRARY_PATH /usr/lib:/usr/openwin/lib:$CDS_INST_DIR/tools/dfII/lib:$CDS_INST_DIR/tools/inca/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/lib:$CDS_INST_DIR/tools/verilog/lib:/usr/dt/lib:/usr/lib/x11:/usr/ucblib:/usr20/dt_cde/lib:/usr/local/lib/gcc-lib:/usr/local/lib:{SILOS}/bin:$UT_ROOT_DIR/PLI

For Undertow versions 1.7 and up, please use the correct PLI according to the simulator type and version. $UT_ROOT_DIR has the following :

libpli.so.nc_verilog *for NC Verilog . This is for CADENCE LDV versions 4.1, 5.1, 5.2 and up*

libpli.so.old_nc_verilog *for CADENCE LDV versions earer than 4.1*

libpli.so.verilog_xl *for Verilog-XL*

Make sure you do the following:

%cd $UT_ROOT_DIR/PLI/

%cp <appropriate libpli.so.#> libpli.so

Also make sure LD_LIBRARY_PATH has $UT_ROOT_DIR/PLI in the path

Add to source code close to top module

initial

begin

$vtDumpvars;

end

In this example, the above has been inserted in 'top.v' file.

If you are simulating your design for the first time, follow these steps:

a) Run "ncprep"

>ncprep -f source

-f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

Note that ncprep will generate the following files and directories.

- *cds.lib*
- *hdl.vars*

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- *INCA_LIB*
- *ncvlog.args*
- *ncelab.args*
- *ncsim.args*

b) Run "ncvlog"

>ncvlog -f ncvlog.arg

c) Add the following line into file "ncleab.args"

-ACCESS +RCW

d) Run "ncelab"

>ncelab -f ncelab.args

e) Run "ncsim"

>ncsim -f ncsim.args

You can then run the simulator again as follows:

./run_ncsim

run_ncsim : script for running all modelsim commands.

---------run_ncsim contents----------

#!/bin/csh -f

Run the NC-Verilog parser (compile the source)

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
ncvlog -f ncvlog.args
if ($status != 0) then
  exit
endif

# Run the NC-Verilog elaborator (build the design hierarchy)
ncelab -f ncelab.args
if ($status != 0) then
  exit
endif

# Run the NC-Verilog simulator (simulate the design)
#ncsim -f ncsim.args +VTCOMPRESS250 +VTVECTORVALUES
ncsim -f ncsim.args
```

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size o

Viewing the NC Sim Waveform

Batch Mode

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

source_code_file is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

ut -v <signal_file>

For example,

ut -v vt.dump

Interactive Mode

a) Cadence Verilog-XL

ut -iv -xl verilog -f <file that lists all source code filenames> -sigfile <signal_filename> -tracefile <trace_filename>

For example,

ut -iv -xl verilog -f source -sigfile fsm.sigs -tracefile fsm.trace

b) Cadence NC Verilog-XL

ut -iv -ncxlmode ncxlmode -f <file that lists all source code filenames> -sigfile <signal_filename> -tracefile <trace_filename>

For example,

ut -iv -ncxlmode ncxlmode -f source -sigfile fsm.sigs -tracefile fsm.trace

c, Cadence NC Verilog

ut -iv -nc ncverilog "-f <file that lists all source code filenames>" -sigfile <signal_filename> -tracefile <trace_filename>

For example,

ut -iv -nc ncverilog "-f source" -sigfile fsm.sigs -tracefile fsm.trace

c, Cadence NC Sim *(Compiled Simulator)*

ut -iv -ncverilog ncsim "-f ncsim.args" -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code file names>"

For example,

ut -iv -ncverilog ncsim "-f ncsim.args" -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

OR

ut -iv -ncverilog ncsim worklib.top:v -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Installing Veritools PLI Routines for Finsim

Sample script for running Finsim

1)Source evironment variables using:

./envsource.

Where envsource is the file containing the set values of all envronment variables

Make sure you set the FINUSERPLIOBJ environment variable follows:

setenv FINUSERPLIOBJ "<path>/vtplifinsim.o <path>/vt_finsimuser.o"

vt_finsimuser.o and vtplifinsim.o are available in our PLI directory. Make sure the path to the PLI directory is specified.

----------envsource file contents----------

setenv FINTRONIC <path where Finsim is installed>

setenv FIN_OBJECT_PATH $FINTRONIC/obj/gcc

setenv FIN_INCLUDE_PATH $FINTRONIC/include

setenv FIN_LIBRARY_PATH $FINTRONIC/lib/gcc

setenv FINTEMPDIR ./fintmp

setenv FIN_COMPILER gcc

#setenv FINUSERPLIOBJ "<path_to>/vtplifinsim.o <path_to>/vt_finsimuser.o"

#NOTE: The path and the files are within "" separated by a space.

setenv FINUSERPLIOBJ "$UT_ROOT_DIR/PLI/vtplifinsim.o $UT_ROOT_DIR/PLI/vt_finsimuser.o"

#The files are available in our PLI directory.

setenv FIN_LICENSE_PATH $FINTRONIC/env

#setenv _ULKFLAGS "-lnsl -lsocket -lintl"

#Above setenv is for SOLARIS ONLY!

setenv FINSYSPLIOBJ $FIN_OBJECT_PATH/verisim.o

setenv $UT_ROOT_DIR <path_to directory holding Undertow distribution>

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

---

2)Add to source code close to top module

initial

begin

$vtDumpvars;

end

In this example, the above has been inserted in 'top.v' file.

3)Modify 'top.cf' file, for example:

-a +vtdbg -ptab vt_ptab

-dsm int -f source

-a : suppresses all warning messages

+vtdbg : generates information for Veritool's source level debugger

-ptab <file> : gets information about pli tasks/functions from the specified ascii file instead of having to link user's pli object files to obtain this information

vt_ptab is available in our PLI directory. Make sure the path to this directory is specified.

-dsm <mode> : sets the default simulation mode to compile (-dsm com) or interpret (-dsm int), -dsm com is the default

Precedence order for simulation mode options

The Verilog compiler determines the simulation mode of a module (compiled or interpreted) based on the invocation options `-comm <mod>', `-intm <mod>', `-comf <file>', `-intf <file>', `-comd <dir>', `-intd <dir>' and `-dsm <mode>'. The precedence order of these options are:

1. -comm <mod>, -intm <mod>
2. -comf <file>, -intf <file>
3. -comd <dir>, -intd <dir>
4. -dsm <mode>

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

For example, if the user types the following:

finvc -intf test.v -comm test

and the module `test' is defined in the file `test.v', then all modules in file `test.v' will be interpreted except `test' which will be compiled.

-f <file> : used to specify file that contains all the user verilog files. Here, 'source' is a file with all of the user's Verilog files(top.v, fsm1.v, fsm2.v, fsm3.v) and iv.v

iv.v is available in the example directory.

NOTE: +VTCOMPRESS250 +VTVECTORVALUES will compress the size of the vt.dump file.Optimization is more in the case of designs with large number of vectors.

4)Run simulator as follows:

./runsim

runsim : script for running all finsim commands.

---------runsim contents----------

#!/bin/csh

 # Usage: runsim <options>

 # This is a sample script which runs the 3 steps required by the

 # FinSim products:

 # 1) finvc --- the Verilog compiler and code generator

 # 2) finbuild --- the simulation builder

 # 3) TOP.sim --- the simulator

 # Different options for these 3 steps can be specified by using the

 # variables FINVCOPT, FINBUILDOPT and FINSIMOPT below. For the compiler

 # finvc, options can also be specified in the command file top.cf or

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
# passed directly in the command line of this script like this:
#             runsim +notimingchecks -dsm int
# For a complete description of the available options please see the User
# Guide or run any of these commands with the option -help.

if (!($?FINTRONIC)) then
  echo ""
  echo " The environment variable FINTRONIC is not set. Please source the file"
  echo " env/env.script in the installation directory and rerun this script"
  echo ""
  exit(1)
endif

set path = ($FINTRONIC/bin/$FIN_COMPILER $path)

# set the options for finvc
set FINVCOPT = "-cf top.cf -a -ol 9"
# set the options for finbuild
set FINBUILDOPT = "-verbose"
# set the options for TOP.sim
set FINSIMOPT = "-verbose"

# run the compiler finvc
finvc $FINVCOPT $*
if ($status != 0) then
  echo ""
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
  echo "***ERROR*** The compiler terminated abnormally. Please check the file finvc.log"
  echo ""
  exit(1)
endif

# run the simulation builder finbuild
finbuild $FINBUILDOPT
if ($status != 0) then
  echo ""
  echo "***ERROR*** The simulation builder terminated abnormally. Please check the files finbuild.log and compile.log"
  echo ""
  exit(1)
endif

# run the simulator TOP.sim
./TOP.sim $FINSIMOPT
if ($status != 0) then
  echo ""
  echo "***ERROR*** The simulation terminated abnormally. Please check the file finsim.log"
  echo ""
  exit(1)
endif

echo ""
echo "The simulation has completed successfully"
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

echo ""

To open the Undertow Suite in batch or interactive mode, the command lines are as follows:

Viewing Finsim Waveform

BATCH MODE

ut -iv -f <source_code_file> -sigfile <signal_file>

<source_code_file> is the file thats lists all the source code files.

For example,

ut -iv -f source -sigfile fsm.sigs

To view just the waveform,

ut -v <signal_file>

For example,

ut -v vt.dump

INTERACTIVE MODE

ut -iv -finsim <simulator_executable> <simulator_options> -sigfile <signal_filename> -tracefile <trace_filename> -ivsimcmp "-f <file that lists all source code

files>"

For example,

ut -iv -finsim TOP.sim -sigfile fsm.sigs -tracefile fsm.trace -ivsimcmp "-f source"

Note: for information about using -tracefile please refer to “Batch Simulation with Command Line” on page 288

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 11

Quick Start: VHDL Example

VHDL Quick Start Simulation Guide

Veritools provides 2 examples of designs for users to run VHDL for ModelTech VHDL simulator and Cadence NC-VHDL. These are found in the following directories:

$UT_ROOT_DIR/FSM_VHDL_NC

FSM_VHDL_NC is a RTL design in VHDL containing 3 state diagrams for Cadence NC-VHDL

$UT_ROOT_DIR/FSM_VHDL_MODELTECH

FSM_VHDL_MODELTECH is a RTL design in VHDL containing 3 state diagrams for Modeltech VHDL simulator.

VHDL Quick Start Guide For NC-VHDL

Access NC-VHDL from your Cadence Installation

To access NC-VHDL, you must set the following environment variable:

On Solaris:

```
setenv LD_LIBRARY_PATH /usr/openwin/lib:/usr/dt/lib:\
$CDS_INST_DIR/tools/lib:${LD_LIBRARY_PATH}
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

To run NC-VHDL, two setup files are required.

1. A cds.lib file. This file contains statements that defines your libraries.

Sample cds.lib:

```
# define library worklib
DEFINE worklib ./worklib
# $CDS_INST_DIR define your NC-VHDL installed directory
SOFTINCLUDE $CDS_INST_DIR/tools/inca/files/cdsvhdl.lib
```

2. An hdl.var file. This file defines which library is the working library.

Sample hdl.var:

```
softinclude $CDS_INST_DIR/tools/inca/files/hdl.var
define LIB_MAP ( $LIB_MAP, + => worklib )
define VIEW_MAP ( $VIEW_MAP, .vhd => vhd)
```

To run NC-VHDL, a library called “worklib” is required.

1. Type the following command to create your library.

```
>mkdir worklib
```

Define Veritools’ vda pli for Cadence NC-VHDL (Dynamic Mode)

On Solaris:

```
setenv LD_LIBRARY_PATH $UT_ROOT_DIR/VDA:${LD_LIBRARY_PATH}
```

Define Veritools’ vda pli for Cadence NC-VHDL (Static Mode)

You have to modify the Cadence NCSIM Makefile to include the Veritools vda pli.

===> define the following into your Makefile

```
CFC_OBJECTS_SAFE=$UT_ROOT_DIR/vda/cfctable.o

FMI_OBJECTS_SAFE= $UT_ROOT_DIR/vda/vdapli.o $UT_ROOT_DIR/vda/fmitable.o
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

where $UT_ROOT_DIR defines your Veritools Installation directory.

Then, type the following command to create your “ncsim” executable.

```
>make ncsim
```

The following is the running example for “make ncsim” for Cadence LDV version 2.2 on Sun Solaris.

```
cc -xs -Bdynamic -o ./ncsim \
      $CDS_INST_DIR/tools/inca/lib/ncsim.o \
      $UT_ROOT_DIR/vda/cfctable.o   \
      $UT_ROOT_DIR/vda/vdapli.o \
      $UT_ROOT_DIR/vda/fmitable.o \
      -L$CDS_INST_DIR/tools/lib -ludm -lsimcontrol -ldataReg_sh -lmpi_sh \
      -lmpsc_sh -lsman_sh $CDS_INST_DIR/tools/virtuos99.01/lib/virtuos.a \
      -L/usr/dt/lib -lXm -L/usr/openwin/lib -lXt -lX11 \
      -L$CDS_INST_DIR/tools/inca/lib -lvoids \
      -L/usr/lib -lsocket -lnsl -ldl -lelf -L/lib -lm -lintl
```

Prepare files for Cadence NC-VHDL

foreign.vhd

Copy the foreign.vhd from $UT_ROOT_DIR/FSM_VHDL_NC to your fsm directory.

where the ‘foreign.vhd’ defines the Veritools’ vda pli

Sample foreign.vhd:

```
entity dump_design is end;
architecture only of dump_design is
  attribute FOREIGN of only : architecture is "Clib:dumpvars";
begin
End;
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Define foreign.vhd within your VHDL design

If you are running your own vhdl design, you have to define the Veritools vda pli within your topmost vhdl module. You don't need to do anything for the fsm design that Veritools gives.

```
-- define the Veritools vda pli component within the architecture
-- definition,
-- just after the "architecture" line
component dump_design end component;


-- define the Veritools vda pli call within your module,
-- just before the "end" line
dump: dump_design;
```

Batch Command Script

Make sure that you have the batch command script called 'batch_cmd'.

Sample 'batch.cmd':

```
run 500000ns

quit
```

Steps to run FSM design example for NC-VHDL

Compiling The FSM design

Batch Mode:

Type the following command to compile the FSM design if you would like to run the design in batch mode.

```
>ncvhdl foreign.vhd fsm1.vhd fsm2.vhd fsm3.vhd top.vhd
```

The following messages should be seen...

```
foreign.vhd:
        errors: 0, warnings: 0
fsm1.vhd:
        errors: 0, warnings: 0
fsm2.vhd:
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
        errors: 0, warnings: 0
fsm3.vhd:
        errors: 0, warnings: 0
top.vhd:
        errors: 0, warnings: 0
WORKLIB.DUMP_DESIGN (entity):
        streams: 1, words: 3
WORKLIB.DUMP_DESIGN:ONLY (architecture):
        streams: 2, words: 26
WORKLIB.FSM1 (entity):
        streams: 6, words: 18
WORKLIB.FSM1:RTL (architecture):
        streams: 3, words: 685
WORKLIB.FSM2 (entity):
        streams: 6, words: 30
WORKLIB.FSM2:RTL (architecture):
        streams: 4, words: 830
WORKLIB.FSM3 (entity):
        streams: 7, words: 48
WORKLIB.FSM3:RTL (architecture):
        streams: 2, words: 641
WORKLIB.TOP (entity):
        streams: 1, words: 3
WORKLIB.TOP:RTL (architecture):
        STD_LOGIC_1164 function "NOT", line 58, is built-in
        STD_LOGIC_1164 function "NOT", line 70, is built-in
        STD_LOGIC_1164 function "NOT", line 74, is built-in
        STD_LOGIC_1164 function "NOT", line 78, is built-in
        STD_LOGIC_1164 function "NOT", line 91, is built-in
        STD_LOGIC_1164 function "NOT", line 102, is built-in
        STD_LOGIC_1164 function "NOT", line 111, is built-in
        STD_LOGIC_1164 function "NOT", line 124, is built-in
        STD_LOGIC_1164 function "NOT", line 138, is built-in
        STD_LOGIC_1164 function "NOT", line 148, is built-in
        10 function(s) built-in
        streams: 10, words: 1840
```

Interactive Mode:

Type the following command to compile the FSM design if you would like to run the design in an interactive mode

```
>ncvhdl -LINEDEBUG foreign.vhd fsm1.vhd fsm2.vhd fsm3.vhd top.vhd
```

Elaborate the FSM design for NC-VHDL

Type the following to elaborate the fsm design.

```
>ncelab -ACCESS +RCW worklib.top:rtl
```

The following messages would be seen.

```
ncelab: v2.2.(p1): (c) Copyright 1995 - 1999 Cadence Design Systems, Inc.
    Elaborating the design hierarchy:
      Foreign entity/architecture elaboration: :TOP(RTL):dump
    Building instance specific data structures.
    Design hierarchy summary:
                    Instances Unique
        Components:         6      5
        Default bindings:   4      -
        Processes:         15     16
        Signals:           21     21
    Writing initial simulation snapshot: WORKLIB.TOP:RTL
```

Running the FSM design in Batch mode

Type the following to run the FSM design in batch mode.

```
>ncsim -TCL top -INPUT batch_cmd
```

The following messages would be seen.

```
ncsim: v2.2.(p1): (c) Copyright 1995 - 1999 Cadence Design Systems, Inc.
ncsim: *W,NOLICC: Unable to checkout license for NC_VHDL_Simulator (2.2), continuing to checkout Affirma_NC_Simulator.
Loading snapshot worklib.top:rtl ..... Done
1
1
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
ncsim> source /opt/LDV22/tools/inca/files/ncsimrc
ncsim> run 500000ns
Ran until 50 US + 0
ncsim> quit
```

Veritools' vda pli options

Define Your DumpFile Name

+VTOPTZ_SFILE<filname|hierarchical_filname>

It indicates dumpfile name. If this option is not specified, a default dumpfile named "vhdl.dump" will be used as the saved fastfile's name.

For Example:

+VTOPTZ_SFILEfsm.dump

+VTOPTZ_SFILE/myhome/project/fsm/fsm.dump

Define Your Dump Module Or Signal Selectively

+DUMPVAR<levels>,<,<module|var>>>*

The first argument indicated how many levels of the hierarchy (below each specified module and instances) to dump. Subsequent arguments specify which scopes of the design to dump. These subsequent argument can specify entire modules or individual variables within a module.

For Example:

+DUMPVAR1,:

Because the first argument is a 1, this invocation dumps all variables within the module top; it does not dump variables in any of the modules instantiated by module top. In NC-VHDL, it treats the toppest module as ":".

+DUMPVAR0,:

Setting the first argument to 0 causes a dump of all variables in the specified module and in all module instances below the specified module. Note that the argument 0 applies only to subsequent arguments that specify module instances, and not to individual variables.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

+DUMPVAR0,:f2,:f1

This call dumps all variables in module /top/f2 and /top/f1.

+DUMPVAR1,:f2,:f1:clock

This call dumps all variables in module /top/f2, along with variable clock in module /top/f1. Note that the argument 1 applies only to the module and not to the individual variable /top/f1/clock.

In NC-VHDL, it treats "/top/f2" as ":f2" and "/top/f1/clock" as ":f1:clock".

VHDL Quick Start Guide For ModelTech VHDL Simulator

Access ModelTech VHDL simulator from your ModelTech Installation

To access ModelTech VHDL VSIM, you must set the following environment variable:

On Solaris:

setenv ModelTech <modeltech_install_dir>/5.X/modeltech/bin

set path = ($path $ModelTech)

Define Veritools' FLI for ModelTech VHDL Simulator

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Prepare files for ModelTech VHDL Simulator

foreign.vhd

Copy the foreign.vhd from $UT_ROOT_DIR/FSM_VHDL_MODELTECH to your fsm directory.

where the ‘foreign.vhd’ defines the Veritools’ FLI

Sample foreign.vhd:

```
entity dump_design is end;

architecture only of dump_design is

        attribute FOREIGN of only : architecture is "dump_design_unit\ $UT_ROOT_DIR/FLI/
fli_modeltech_53.so";

begin

End;
```

Define foreign.vhd within your VHDL design

If you are running your own vhdl design, you have to define the Veritools vda pli within your toppest vhdl module. You don’t need to do anything for the fsm design that Veritools gives.

```
-- define the Veritools vda pli component within the architecture
-- definition,
-- just after the "archiecture" line
component dump_design end component;


-- define the Veritools vda pli call within your module,
-- just before the "end" line
dump: dump_design;
```

Steps to run FSM design example for ModelTech VHDL Simulator

1. **cd <*veritools_install_dir*>**
2. **cp -pR ./FSM_VHD <*my_suitable_work_dir*>/FSM_VHDL**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

3. Ensure that the following environment variables are set.
   - **setenv ModelTech <*modeltech_install_dir*>/5.X/modeltech/bin**
   - **setenv PLIOBJS $UT_ROOT_DIR/FLI/vtpli_modtech.so**
   - **set path = ($path $ModelTech)**
4. Verify that valid ModelTech software licenses are available.

Creating Working Library

1. To create a working directory for ModelTech, type:

   vlib work

Compiling The FSM Design

1. To compile the *dump_design* Entity and Architecture, type:

   vcom foreign.vhd fsm1.vhdl fsm2.vhdl fsm3.vhdl top.vhdl

You should see the following messages:

```
Model Technology ModelSim SE/EE vcom ... –
Loading package standard –
Loading package std_logic_1164 --
Compiling entity dump_design –
Compiling architecture only of dump_design
Compiling entity fsm1 –
Compiling architecture rtl of fsm1 –
Compiling entity fsm2 –
Compiling architecture rtl of fsm2 –
Loading package numeric_std –
Compiling entity fsm3 –
Compiling architecture rtl of fsm3 –
Compiling entity top –
Compiling architecture rtl of top –
Loading entity fsm1 –
Loading entity fsm2 –
Loading entity fsm3 –
Loading entity dump_design
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Running FSM Design In Batch Mode

Type the following to run the FSM design in batch mode.

```
>>vsim -c -quiet -do ’run 500000; quit;’ top
```

The following messages would be seen.

```
Reading <modeltech_install_dir>/bin/../sunos5_251/../tcl/vsi m/pref.tcl


# 5.4
# vsim -c top
# Loading <modeltech_install_dir>/bin/../sunos5_251/../std.standard
# Loading <modeltech_install_dir>/bin/../sunos5_251/../ieee.std_logic_1164(body)
# Loading
<modeltech_install_dir>/bin/../sunos5_251/../ieee.numeric_std(body)
# Loading work.top(rtl)
# Loading work.fsm1(rtl)
# Loading work.fsm2(rtl)
# Loading work.fsm3(rtl)
# Loading work.dump_design(only)
# Loading <veritools_install_dir>/FLI/fli_modtech_53.so
VSIM 1>run 500000
VSIM 2>quit
```

Veritools’ FLI options

Define Your DumpFile Name

+VTOPTZ_SFILE<filname|hierarchical_filname>

It indicates dumpfile name. If this option is not specified, a default dumpfile named "vhdl.dump" will be used as the saved fastfile’s name.

For Example:

+VTOPTZ_SFILEfsm.dump

+VTOPTZ_SFILE/myhome/project/fsm/fsm.dump

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Define Your Ignored Module

+IGNORE_MODULE<hierarchical_instance_name|module_name|hierarchical_module_name>

It specifies a module which will be ignored from the dump. All instance within the ignored module as well as the ignored module will not be dumped to the vhdl saved fastfile.

For Example:

+IGNORE_MODULEtop <---- module_name

+IGNORE_MODULEf1 <---- module_name

+IGNORE_MODULE/top/f1 <---- hierarchical_module_name

Note that only one module could be defined.

Define Your Dump Module/Signal Selectively

+DUMPVAR<levels>,<,<module|var>>>*

The first argument indicated how many levels of the hierarchy (below each specified module and instances) to dump. Subsequent arguments specify which scopes of the design to dump. These subsequent argument can specify entire modules or individual variables within a module.

For Example:

+DUMPVAR1,top

Because the first argument is a 1, this invocation dumps all variables within the module top; it does not dump variables in any of the modules instantiated by module top.

+DUMPVAR0,top

Setting the first argument to 0 causes a dump of all variables in the specified module and in all module instances below the specified module. Note that the argument 0 applies only to subsequent arguments that specify module instances, and not to individual variables.

+DUMPVAR0,/top/f2,/top/f1

This call dumps all variables in module /top/f2 and /top/f1.

+DUMPVAR1,/top/f2,/top/f1/clock

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This call dumps all variables in module /top/f2, along with variable clock in module /top/f1. Note that the argument 1 applies only to the module and not to the individual variable /top/f1/clock.

Batch Simulation of FSM Example

Run Batch Simulation

You should follow the previous sections to generate simulation dump files from ModelTech VHDL simulator.

Viewing Result By Using Undertow Waveformer

1. To view simulation signal/wave results, type:

   ut -v vhdl.dump

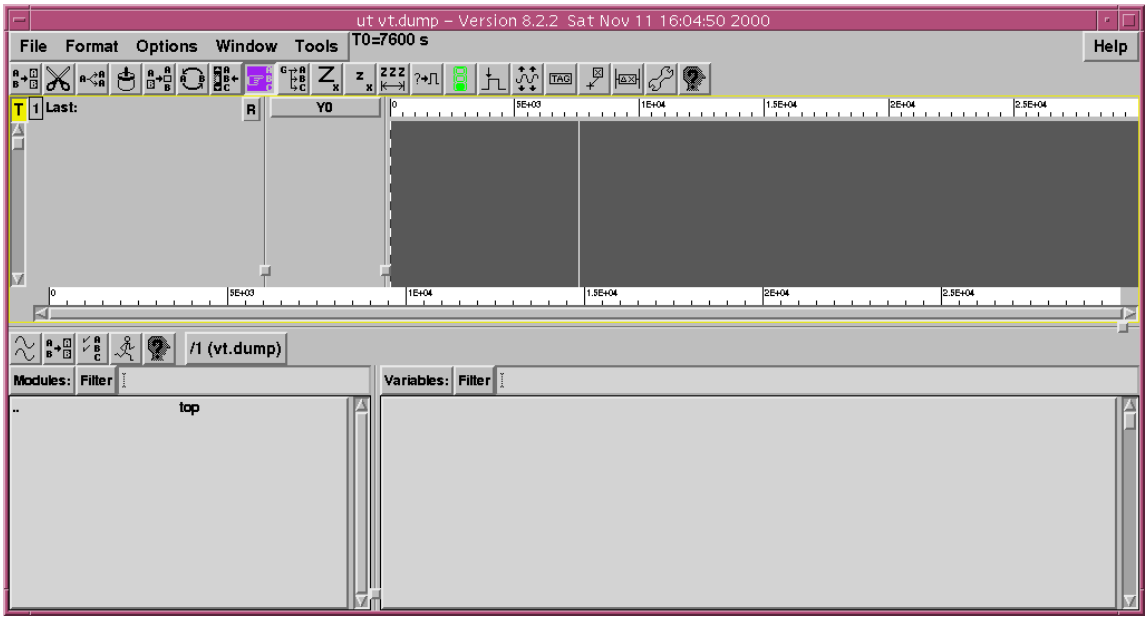
   The Undertow signal/waveform viewer appears on screen.

2. Locate the **Chooser** or **Hierarchical Browser** icon.

   The icon is purple, and depicts a hand pointed at the letters **ABC**.

3. Use the left mouse button to select and open the *browser*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



In the Browser:

1. Select **Display** and **Analyze** signals.
2. To select the *top* module, click the **left** mouse button on the word **top**.

- The **top** module name disappears.
- The signals for this level of hierarchy appear in the signal selector area (the area to the right of the hierarchical browser), and under the *Variables* button.

3. To select all of the signals in the *signal selector area*, left-click on the **Display** icon, and then click the **All** icon to the right.

   These two icons are at the upper left side of the *hierarchical browser*.

   - **Display** appears as a pair of sine waves.
   - **All** is to right, and appears as a pair of **A B** block characters.

   The Undertow window shows all of the signals at this **top** level of hierarchy.

The Undertow window includes **three time cursors** to analyze signals:

1. Place the mouse in the center of the waveforms, and hold down the **middle** mouse button.

   This is the **t0** cursor.

 Copyright Veritools 2004 All Rights Reserved

2. Move the mouse around.
   - The time value changes, at the top of the Undertow window.
   - Signal value changes on the left.
3. To **drop** the cursor at a particular time, **release** the **middle** mouse button.
4. To place the **t1** cursor, use the **middle** mouse button while holding down the **Shift** key.
5. To place the **t2** cursor, use the **middle** mouse button while holding down the **Ctrl** key.

To **Zoom**, you can be use either the icons at the top of Undertow window, or the **left** mouse, **left** mouse, **right** mouse sequence.

1. Use the **left** mouse to place the **first zoom cursor**.
2. Use the **left** mouse place the **second zoom cursor**.
3. Use the **right** mouse click to **zoom** the Undertow window to boundaries of the zoom cursors.

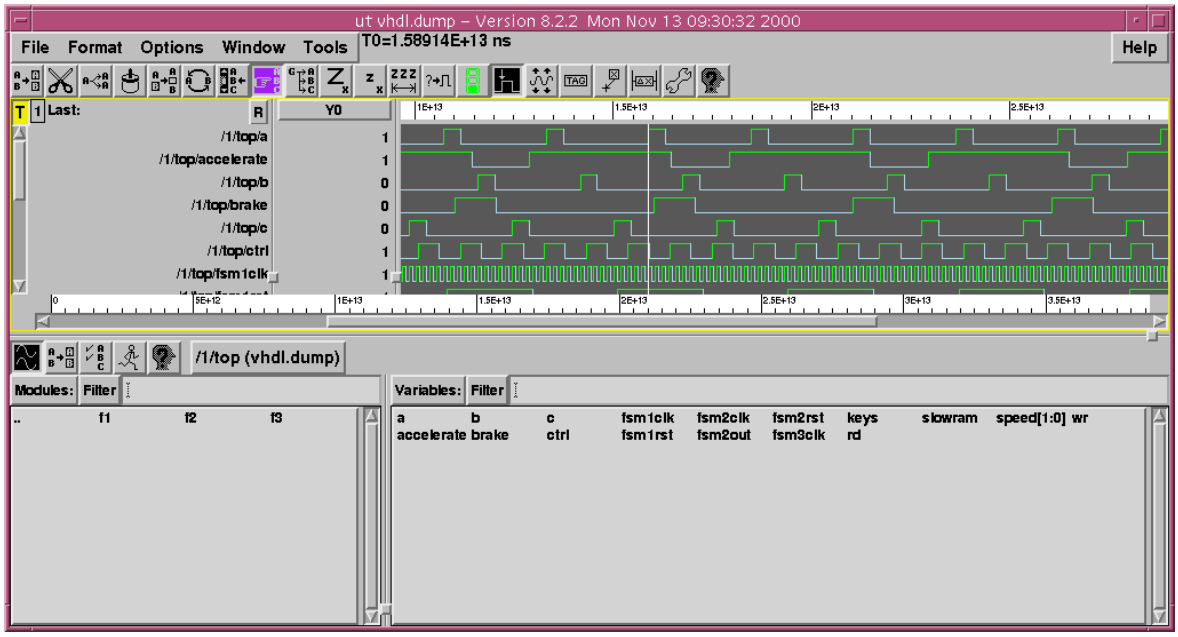
   *Note: If you hold down the **right** mouse button and **drag** the mouse over an area, then **releasing** the mouse zooms into that area.*

To **pan**:

1. Hold down the **left** mouse or **right** mouse buttons, in the **slide window** (at the bottom of the Undertow window).

   Holding down the **left** mouse **pans left**, and the **right** mouse button **pans right**.
2. To control the panning speed, move the mouse to the left for a slower pan, or to the right for a faster pan, while you hold down the left or right mouse button.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Undertow Suite Simulation of FSM For Cadence NC-VHDL Simulator

Undertow Suite Batch Simulation

It is recommended to perform the above “Undertow Batch” demonstration to ensure that the Cadence NC-VHDL and Veritools tools are up and running.

Invoking Undertow Suite By Using A Single Command

1. To ensure that the ModelTech and Veritools tools are working properly, perform the “Undertow Batch” demonstration.

   See Section , “Undertow Suite Batch Simulation”.

2. To start an Undertow Suite Interactive simulation, using the ModelTech VHDL simulator, type the following command:

   ut -iv -vhdl -sigfile fsmvhdl.sigs -ivsimcmp “-f fsm_vhdl.vc”

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The Undertow Suite **Source** and **Undertow** windows open, with the **Source** window at the top.

3. In the **Source** Window, use the left mouse button to select the following menu item:

   Session => Source Code Files/Simulator Options

   The **Simulator Options** window pops up.

   - *ModelTech VHDL - Optimized* file is selected.
   - A complete path to the ModelTech **vsim** is used.
   - **Simulator Compile Options** lists the names of the source files.
   - **Simulator Options** lists the **top** module.

Invoking Undertow Suite By A Project File

Begin an Undertow Suite Batch simulation with Cadence NC-VHDL by typing the following command:

ut -iv -prj fsm.prj

The Undertow Suite “Source” and “Undertow” windows should appear with the “Source” window at the top.

Sample fsm.prj:

```
ivHierFile          =
verExecutable       = ncsim
executeDirectory    =
dumpFileName        = fsm.sigs
traceFileName       =
connFileName        = fsm.conn
enableMonitorAll    = False
ncvhdl              = True
fastModeFlush       = 60
simCmdLine          = top
simCompileLine      = top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl
```

Invoking Undertow Suite Batch Simulation

Now, locate and use the **left** mouse to open the “Actions” menu and select “Load Batch Run”.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Actions => Load Batch Run

Undertow Suite Interactive Simulation

It is recommended to perform the above “Undertow Batch” demonstration to ensure that the ModelTech and Veritools tools are up and running.

Invoking Undertow Suite By Using A Single Command

Begin an Undertow Suite Interactive simulation with ModelTech's VHDL simulator by typing the following command:

ut -iv -vhdl -sigfile fsmvhdl.sigs -ivsimcmp “-f fsm_vhdl.vc”

The Undertow Suite “Source” and “Undertow” windows should appear with the “Source” window at the top.

Locate the Source Window and then use the left mouse to open the “Session” menu, and select “Simulator...”

Session => Source Code Files/Simulator Options

The “Simulator Options” pop-up window should appear. Note; that "NC-VHDL is selected, a complete path to the ‘ncsim’ executable is used, the names of the source files are under Simulator Compile Options, and the “top” module is under Simulator Options.

Invoking Undertow Suite By A Project File

Begin an Undertow Suite Interactive simulation with Cadence NC-VHDL by typing the following command:

ut -iv -prj fsm.prj

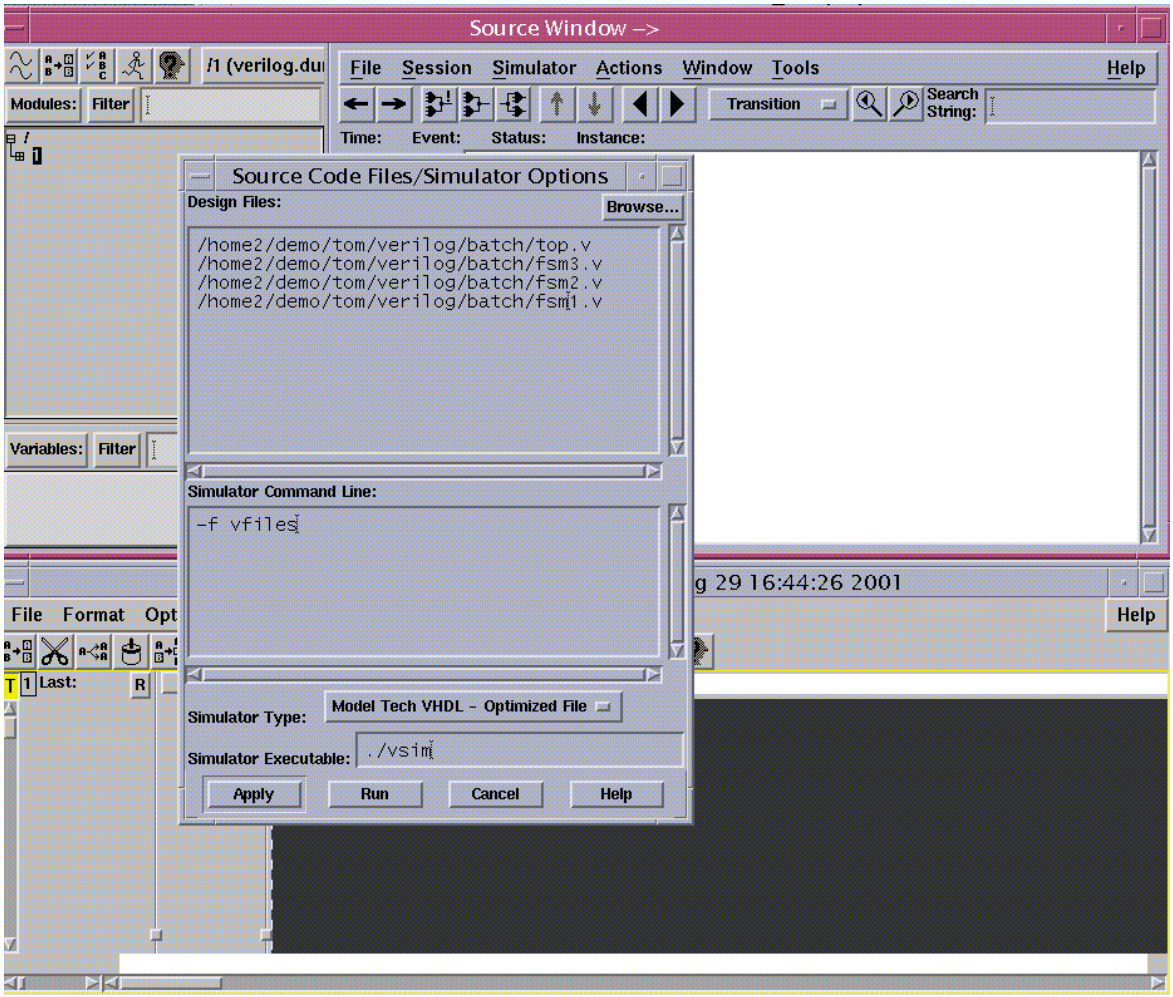
Sample fsm.prj:

```
ivHierFile          =
verExecutable       = ncsim
executeDirectory    =
dumpFileName        = fsm.sigs
traceFileName       =
connFileName        = fsm.conn
enableMonitorAll    = False
ncvhdl              = True
fastModeFlush       = 60
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

simCmdLine = top

simCompileLine = top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl

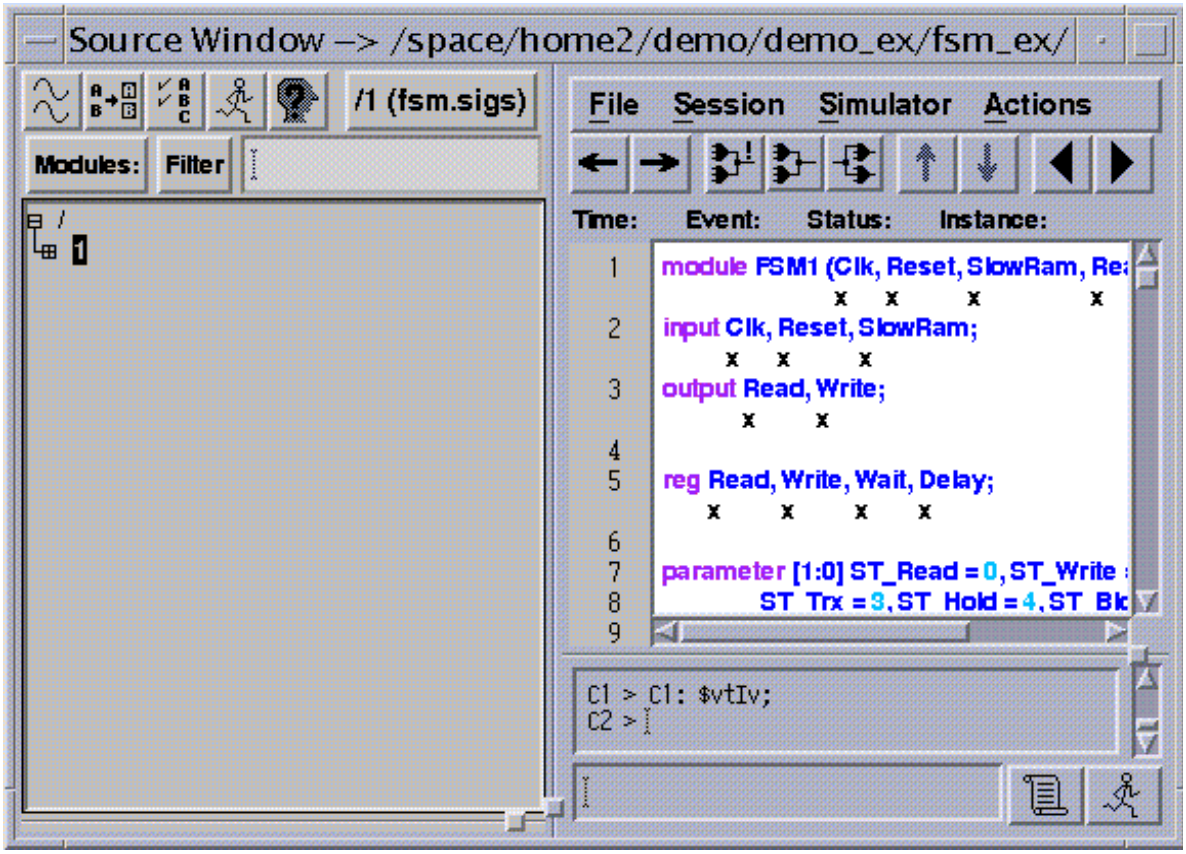


Now edit the complete path to ncsim to point to the vsim executable at this location site. This STEP MUST TAKE PLACE for a successful interactive simulation. Close the Simulator Options pop-up window by clicking “ok”, after editing the ncsim path.

Now, locate and use the **left** mouse to open the “Simulator” menu and select “Run”.

Simulator => Run

The “Source” window should fill in with source code, and the “**NCSIM** >” prompt should been seen directly below the “Source” window.

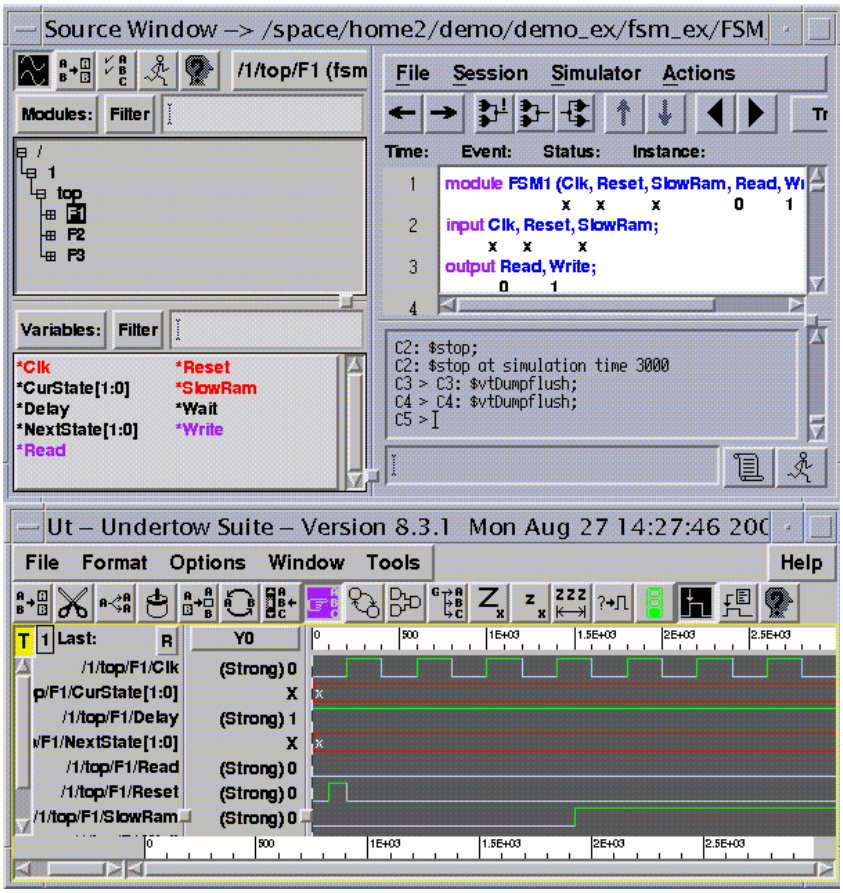

Source Window –> /space/home2/demo/demo_ex/fsm_ex/
/1 (fsm.sigs)
File Session Simulator Actions
Modules: Filter
Time: Event: Status: Instance:
module FSM1 (Clk, Reset, SlowRam, Re
input Clk, Reset, SlowRam;
output Read, Write;
reg Read, Write, Wait, Delay;
parameter [1:0] ST_Read = 0, ST_Write
ST_Trx = 3, ST_Hold = 4, ST_Bk
C1 > C1: $vtIv;
C2 >

Now, locate the “Hierarchical Module Browser” to the left of the “Source Window”. Use the left mouse to click on the + sign next to the number **1** to expand hierarchy. Then use the left mouse to expand the **Top** hierarchy, and then use it to highlight the **F1** module.

The “Variable” window, directly below the “Hierarchical Module Browser” should fill with color coded input(Red)/output(Blue)/internal(Black) signal names.

Now, locate the **Display** and **All** icons at the extreme top left of the “Hierarchical Module Browser”. The **Display** icon is depicted by two sine waves, use the **left** mouse and click on it. The **All** icon is to the right of the **Display** icon, and depicted **AB** block letters use the left to click on it.

The Undertow Window should display all of the F1 modules signal names with their initial signal values.



Now, locate the **Run Simulation** icon. It is located at the extreme bottom right of the "Source Window" and is depicted as a running stick figure, use the left mouse button to open the icons menu and select a simulation run of 100000 time units.

Run Simulation => go 100000

The "Undertow" window should fill in with 100000 time units of signal information, and the ">" is ready for another simulation command.

Use the "Simulator" menu to end simulation.

Simulator => Kill

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Undertow Suite Simulation of FSM For ModelTech VHDL Simulator

Undertow Suite Batch Simulation

It is recommended to perform the above “Undertow Batch” demonstration to ensure that the ModelTech VHDL Simulator and Veritools tools are up and running.

Begin an Undertow Suite Batch simulation with ModelTech VHDL Simulator by typing the following command:

ut -iv -vhdl -sigfile fsmvhdl.sigs -ivsimcmp “-f fsm_vhdl.vc”

The Undertow Suite “Source” and “Undertow” windows should appear with the “Source” window at the top.

Locate the Source Window and then use the left mouse to open the “Session” menu, and select “Simulator...”

Session => Source Code Files/Simulator Options

The “Simulator Options” pop-up window should appear. Note; that "ModelTech VHDL Optimized File is selected, a complete path to the ModelTech vsim is used, the names of the source files are under Simulator Compile Options, and the “top” module is under Simulator Options.

Now, locate and use the **left** mouse to open the “Actions” menu and select “Load Batch Run”.

Actions => Load Batch Run

Undertow Suite Interactive Simulation

It is recommended to perform the above “Undertow Batch” demonstration to ensure that the ModelTech and Veritools tools are up and running.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Invoking Undertow Suite By Using A Single Command

Begin an Undertow Suite Interactive simulation with ModelTech's VHDL simulator by typing the following command:

ut -iv -vhdl -sigfile fsmvhdl.sigs -ivsimcmp “-f fsm_vhdl.vc”

The Undertow Suite “Source” and “Undertow” windows should appear with the “Source” window at the top.

Locate the Source Window and then use the left mouse to open the “Session” menu, and select “Simulator...”

Session => Source Code Files/Simulator Options

The “Simulator Options” pop-up window should appear. Note; that "NC-VHDL is selected, a complete path to the ‘ncsim’ executable is used, the names of the source files are under Simulator Compile Options, and the “top” module is under Simulator Options.

Invoking Undertow Suite By A Project File (FLI Mode)

Begin an Undertow Suite Interactive simulation with Cadence NC-VHDL by typing the following command:

ut -iv -prj fsm.prj

Sample fsm.prj:

```
ivHierFile          =
verExecutable       = ncsim
executeDirectory    =
dumpFileName        = fsm.sigs
traceFileName       =
connFileName        = fsm.conn
enableMonitorAll    = False
modeltech           = False
modeltechvhdl       = True
fastModeFlush       = 60
simCmdLine          = top
simCompileLine      = top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Invoking Undertow Suite By A Project File (VCD Mode)

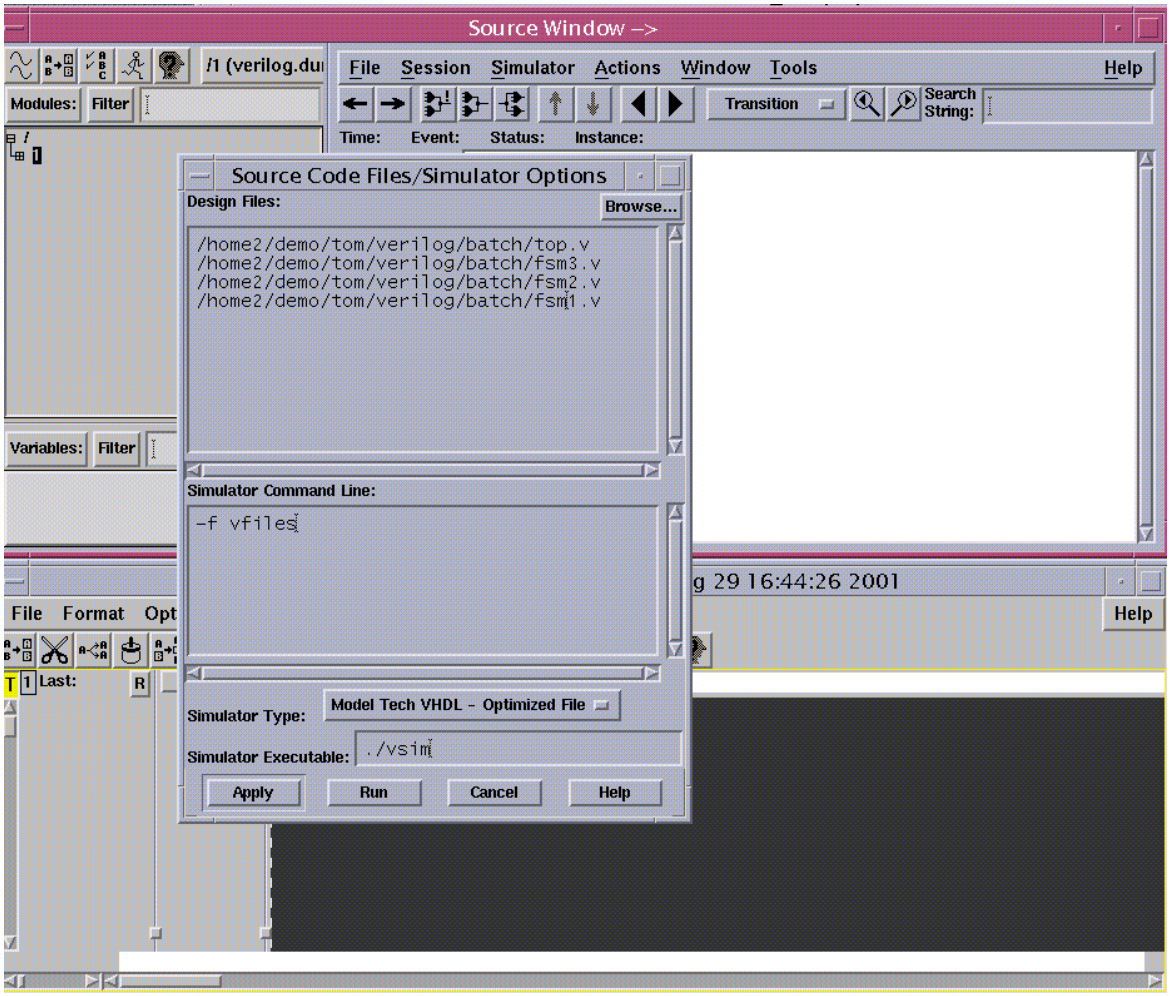
Begin an Undertow Suite Interactive simulation with Cadence NC-VHDL by typing the following command:

ut -iv -prj fsm.prj

Sample fsm.prj:

```
ivHierFile          =
verExecutable       = ncsim
executeDirectory    =
dumpFileName        = fsm.sigs
traceFileName       =
connFileName        = fsm.conn
enableMonitorAll    = False
modeltech           = False
modeltechVhdlOpt    = True
modeltechVhdlVcd    = True
fastModeFlush       = 60
simCmdLine          = top
simCompileLine      = top.vhdl fsm1.vhdl fsm2.vhdl fsm3.vhdl
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



4. Edit the complete path to **vsim**, to point to the **vsim** executable at this location site.

   Note: For a successful interactive simulation, you MUST complete this step.

5. To close the **Simulator Options** pop-up window, click OK, after you edit the **vsim** path.

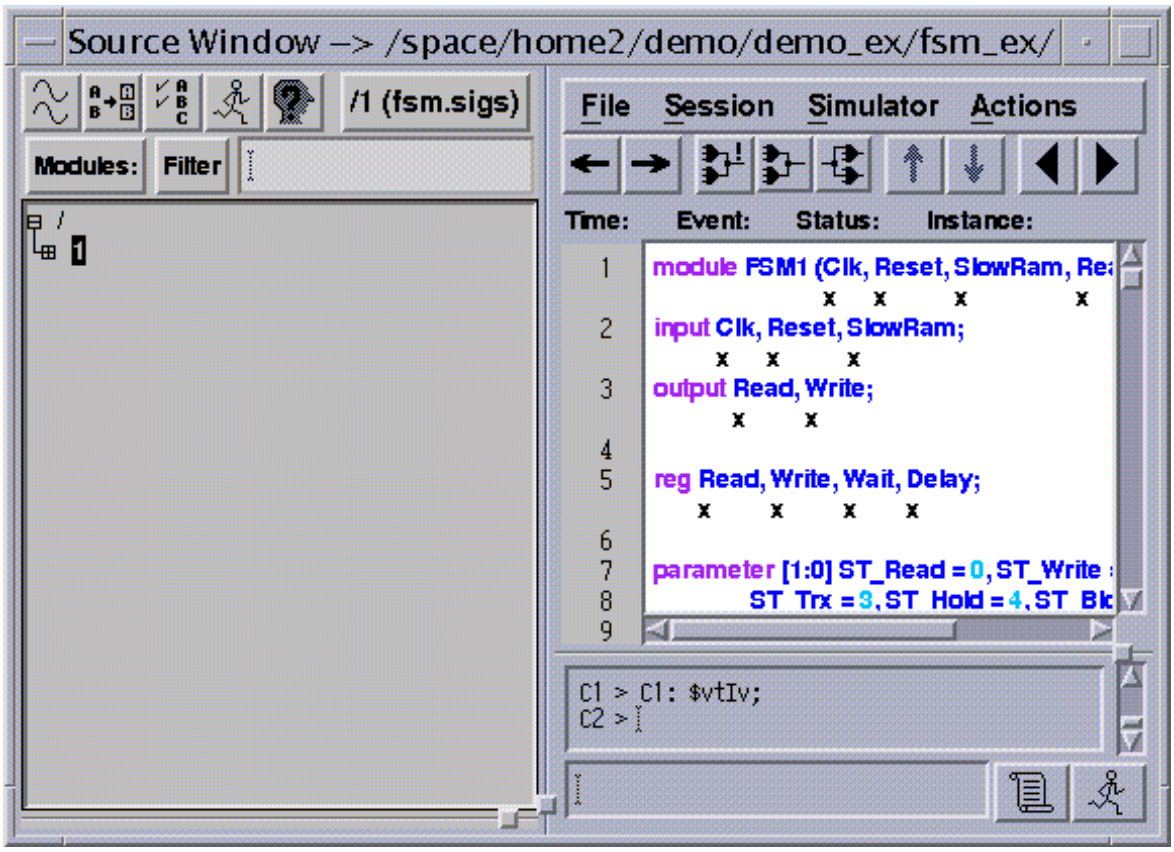
6. Use the **left** mouse button to open the following menu item:

   Simulator => Run

   - The source code appears in the **Source** window.
   - The **VSIM >** prompt appears directly below the **Source** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

0.0.0.0.a Expanding the Hierarchy



1. In the **Hierarchical Module Browser** (to the left of the **Source** Window), left-click on the + sign next to the number **1**, to expand the hierarchy.

2. Use the left mouse to expand the **Top** hierarchy, and then use it to highlight the **F1** module.

   The **Variable** window, directly below the **Hierarchical Module Browser**, fills with color-coded signal names:

   • input(Red)

   • output(Blue)

   • internal(Black)

   The **Display** and **All** icons appear at the extreme top left of the **Hierarchical Module Browser**.:

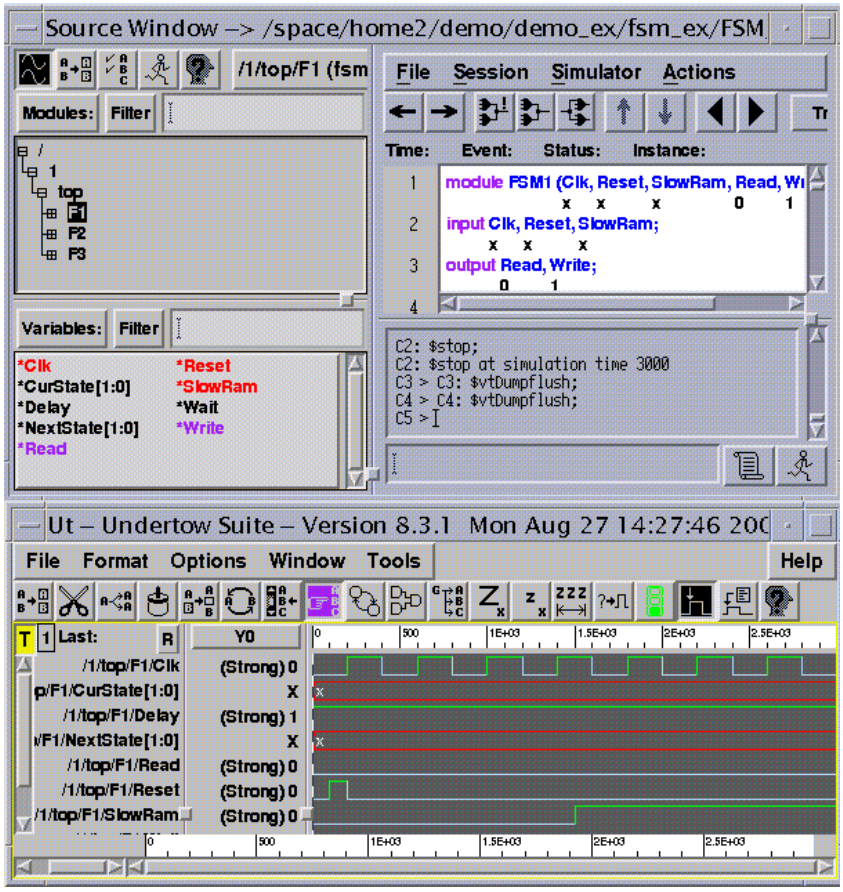
   • The **Display** icon depicts two sine waves.

   • The **All** icon is to the right of the **Display** icon, and depicts **AB** block letters.

3. Use the left mouse button to click on both the **Display** and **All** icons.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

The **Undertow** Window displays all of the signal names in the **F1** module, and their initial signal values.



0.0.0.0.b Running the Simulation

The **Run Simulation** icon is located at the extreme bottom right of the **Source** Window. It shows a running stick figure.

1. Use the left mouse button to open the icons menu, and select a simulation run of 100000 time units.

   Run Simulation => go 100000

   • The **Undertow** window displays 100000 time units of signal information.

   • The > prompt is ready for another simulation command.

2. Use the **Simulator** menu to end simulation:

   Simulator => Kill

General Guide Within Undertow Suite Simulation

Working With Wave Forms In Undertow Suite.

The Undertow window uses **three time cursors** to analyze signals.

1. Place the mouse in the center of the waveforms, and hold down the **middle** mouse button.

   This is the **t0** cursor.

2. Move the mouse around.

   - The time value changes, at the top of the **Undertow** window.
   - The signal value also changes, on the left.

3. **Release** the **middle** mouse button, to **drop** the cursor at a particular time.

4. To place the **t1** cursor, use the **middle** mouse button while holding down the **Shift** key.

5. To place the **t2** cursor, use the **middle** mouse button while holding down the **Ctrl** key.

To **Zoom**, you can be use either the icons at the top of Undertow window, or the **left** mouse, **left** mouse, **right** mouse sequence.

1. Use the **left** mouse to place the **first zoom cursor**.

2. Use the **left** mouse place the **second zoom cursor**.

3. Use the **right** mouse click to **zoom** the Undertow window to boundaries of the zoom cursors.

   *Note: If you hold down the **right** mouse button and **drag** the mouse over an area, then **releasing** the mouse zooms into that area.*

SCHEMATIC WINDOW

1. To view the **Schematic** window from the Undertow Suite, select ***Window => Tile Windows => Schematic***.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

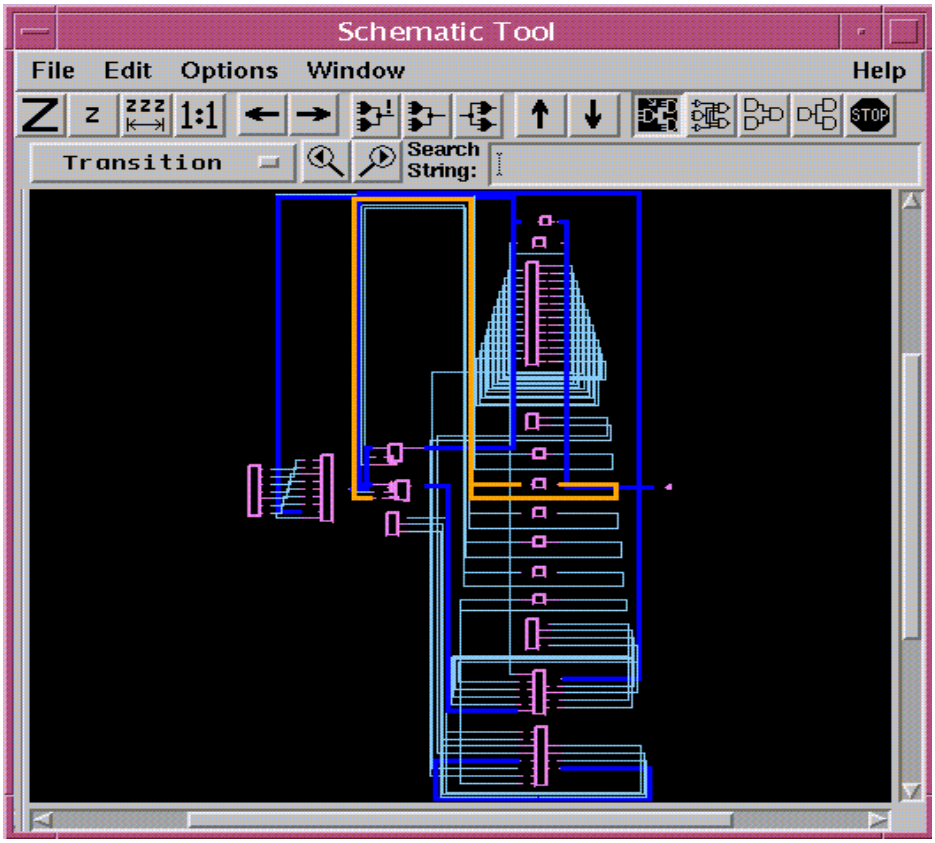
2. Use the middle mouse button to drag and drop the **top** module, from the **source code** browser (the area on the far left of the source code window), to the **Schematic** window.

3. To see the schematic (for the first-level down from the top module), double-click the left mouse button, in the **top** module.

4. To see the entire schematic at this level, select ***Options*** => ***Show Entire Schematic***.

5. Click **Highlighted Net**.

   When you drag-and-drop any signal, from any number of windows in **Undertow**, the **Schematic** window shows where this net is located.



6. Drag and drop any hierarchy block to the **Schematic** window.

7. Double-click inside of the module element.

   The RTL schematic, for the first level down in this module, appears.

8. Left-click on any block in this schematic window.

   The contents of this element appear in the **Source Code** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

9. If Highlight Net is toggled on (Step 5), you can select any signal or block from:
   - The **source code** window, or
   - The **signal selector**, or
   - Undertow.
10. Drag the selected signal onto the **schematic** window.

    You can see how and where the signal connects to the other loads and drivers in your design.

STATE DIAGRAM WINDOW

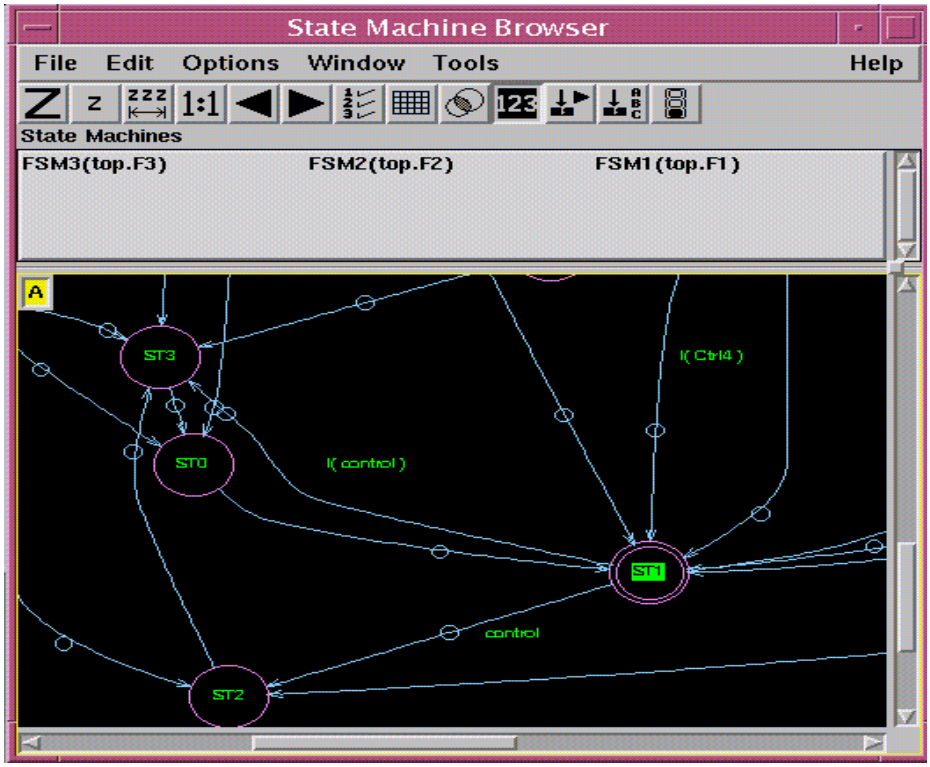
1. To view the **State Diagram** window from the Undertow Suite, select ***Window*** => ***Tile Windows*** => ***State***.

   In the **State Diagram** window, the top window is a state diagram selector.
2. In the state diagram selector window, click on **FSM2**.

   The state diagram opens, for the **FSM2** module.
3. To zoom in this state diagram, drag the right mouse button over a small area of the state diagram.
4. Click the **Step Forward** or **Step Backward** icon in the *state diagram* window,
   - The state diagram dynamically moves forward or backward.
   - **Undertow** automatically remains synchronized with the state that you stepped to.
   - The Undertow **Waveform** window also displays the state variable, with the state values on it.
5. Use any of the state diagram tools to analyze this state diagram, such as:
   - Persistence.
   - Coincidence.
   - Show State Table.
6. Drag any state from the **State Diagram** window, to the **Source Code** window.

   You can see *exactly* where this state is located in the source code.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Now to run through all of the windows of Undertow Suite do the following;

Step the State Diagram to any point using the Step Forward, and Step Backward icons on the State Diagram window.

Drag the active state (shown with a double bubble), to the source code window.

To open and use all of the windows in Undertow Suite, do the following:

1. Step the **State Diagram** to any point, using the *Step Forward*, and *Step Backward* icons on the **State Diagram** window.

2. Drag the active state (shown with a double bubble), to the **source code** window.

   The source code window shows where this state is assigned to the *Next State*.

3. Drag *Next State* to the **Schematic** window.

   The Next State wire is highlighted.

4. Drag the highlighted *Next State* wire to the signal name area, in the Undertow **waveform** window.

   The waveform displays for *Next State*.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

5. From the **Schematic** window, also drag-and-drop *Current State* onto the **Waveform** window.

   Current State is the latched state variable, which is clocked from *Next State*.

6. To select the *Current State* signal on the **waveform** window, click the left mouse button on the **GoTo** icon in the Undertow window.

7. Click the **GoBack** button, then the **GoTo** button.

   The **State Diagram** now steps both forward and backward, exactly like using the *Forward* and *Backward* Step icons on the **State Diagram** window.

simulator first.

1. Specify the path to the simulator executable.

2. Add simulation files.

3. Select browse.

   You can load individual files, or if you have a list of files with valid verilog options in it, you can select just that.

4. Click apply.

   This loads the file into the design files window.

   *Note: If you are using a single file that lists files and options add a '**-f**' and space in front of the path to the file.*

5. Add simulator-specific options to the simulator command line.

   For MTI it might be something like '-c top' .

6. Click Apply.

   The netlist loads into the Source Window.

7. Use Simulator ->Run to start the simulation.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

CHAPTER 11

Undertow Integration Guide With Third-Party Tools

Introduction

The Undertow-DFII Integration kit integrates the Undertow *waveform* viewer into Cadence dfII. Cadence users can use this integration kit, to probe schematics and text in the Undertow *waveform* viewer; this integration helps you to debug your designs within the Cadence Composer environment. Before you can use this integration, you must define a successful OSS netlist directory, within the integration environment, for a target digital or analog simulator.

Veritools has successfully tested, and fully supports, integration with the following target simulators:

- Cadence Verilog-XL, NC-Verilog
- Antrim

Undertow-DFII Integration Kit

The integration kit includes the following two files:

Undertow.Init:

```
unless( boundp('Undertow_loaded)
    Undertow_loaded = t
    cond(
        ( (simUndertowSeDir = getShellEnvVar("simUndertowSeDir"))
          && stringp(simUndertowSeDir)
          printf("Undertow: simUndertowSeDir=%s\n" simUndertowSeDir)
        )
        ( t
            simUndertowSeDir = prependInstallPath( "local/undertow" )          )
    ) ;cond
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
 if( isDir(simUndertowSeDir)
then
   load( strcat( simUndertowSeDir "/undertow.ile"))
   Undertow_loaded = simUTConfig()
else
   println("ERROR - environment variable 'simUndertowSeDir' is not
            set properly, not able to load
            undertow.ile from %s" simUndertowSeDir)
    Undertow_loaded = nil
) ; endif
) ;unless
```

Append this file into your Cadence .cdsinit file, so that dfII downloads the Undertow-dfII Skill automatically.

undertow.ile:

This is the encrypted Skill for the Undertow-dfII UI integration environment.

Installation

1. Create a new directory:

   <*cds_installed_hierarchy*>/local/undertow

2. Place the undertow.ile file under this new directory.

Environment Setup

1. Set the following for Undertow, in your .cshrc file:

   setenv UT_ROOT_DIR undertow_install_directory_path

   setenv UT_WORK_DIR undertow_working_directory_path

   setenv XENV $UT_ROOT_DIR/XENV

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Append the Undertow.init file into your .cdsinit file, to make sure that the Undertow encrypted Skill file loads automatically, whenever you start dfII.

3. Add the following line into your verilogI.ini file for your Undertow-Verilog Integration environment.

   Note: Add this line to the bottom of a Skill procedure named vlogifCreatePW().

   Using this additional Skill call, the Verilog-XL Integration Encapsulation window automatically loads the **Undertow** menu into the banner menus.

   The "verilogI.ini" should be located under your "<cds_install_hierarchy>/tools/dfII/etc/context". Please ask your CAD engineer for help if you don't have permission to modify the file.

   "verilogI.ini"

   .....

   procedure(vlogifCreatePW(@optional (winCoord nil))

   .....

   veritoolVlogAddUT()

   return(t)

   .....

)

   .....

Standalone Usage Of Undertow-Cadence DFII Integration

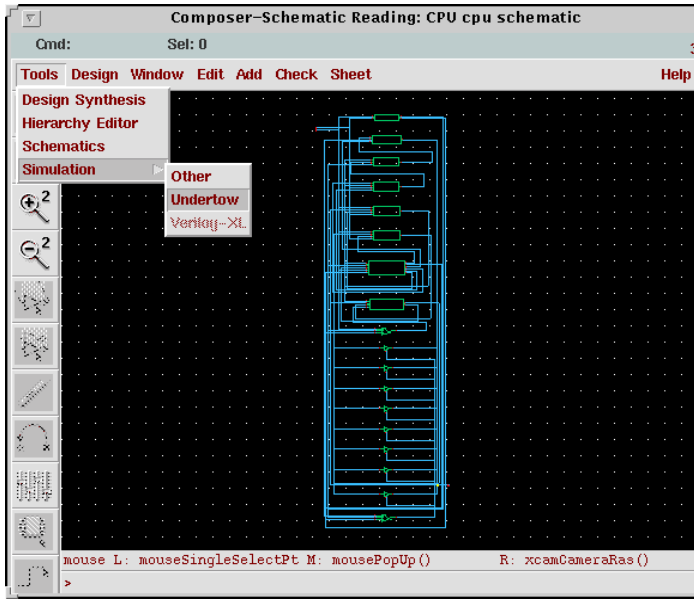
This section describes how to use Undertow to display signals in a stand-alone Cadence Composer environment.

Usage

1. Open a schematic or Verilog text cell, from either the Library Manager or the ***CIW*** => ***File*** => ***Open*** menu item.

2. Select the ***Tools*** => ***Simulation*** => ***Undertow*** menu item.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



A form opens, and asks you to specify the name of a pre-existing **OSS Simulation Run** directory.

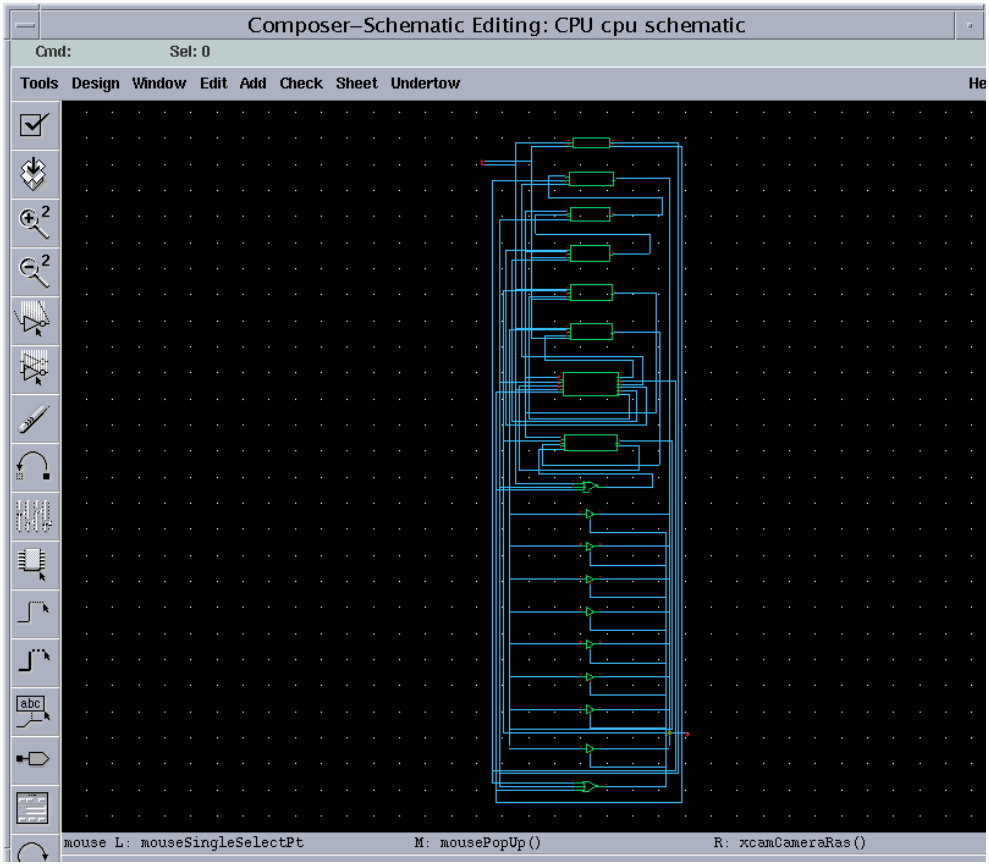
3. If you have not already done so, create the OSS run directory.
4. Make sure that both the results, and the netlist, that the netlist Integration environment generated for the target simulator, are in this directory.

This release supports both **Buses** netlist and **Bit** netlist modes for the Verilog Integration environment.

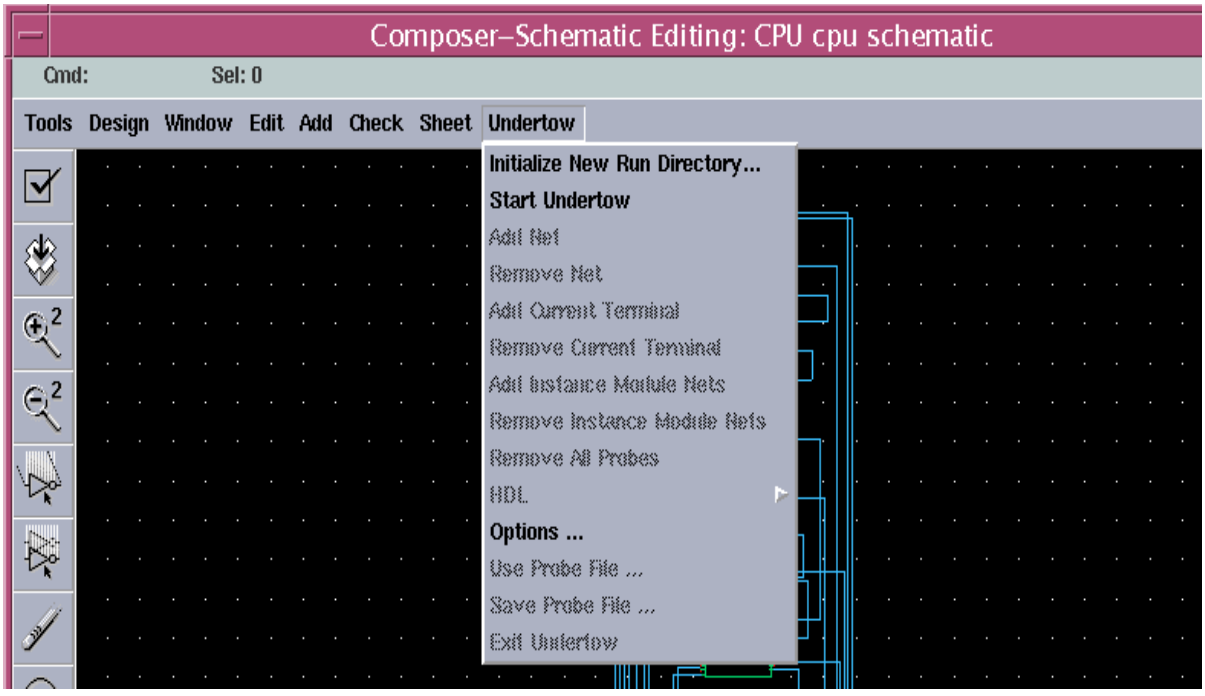
Two new menus appear in the banner menus of either the Composer schematic window, or the Verilog-HDL window:

- **Run directory**
- **Undertow**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



The **Undertow** menu contains the following options:

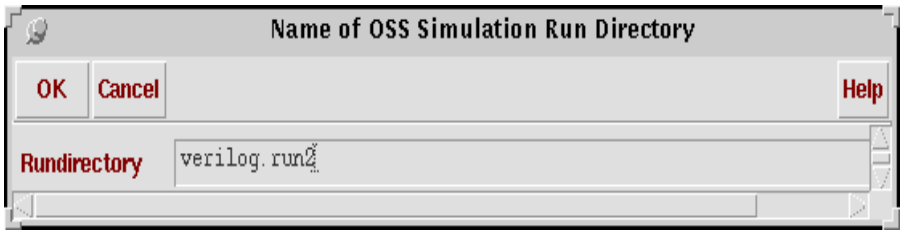


- **Initialize New Run Directory ...**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This menu item pops up the **OSS Simulation Run Directory** form.

Use this form to switch to a new pre-existing **OSS Simulation Run** directory.



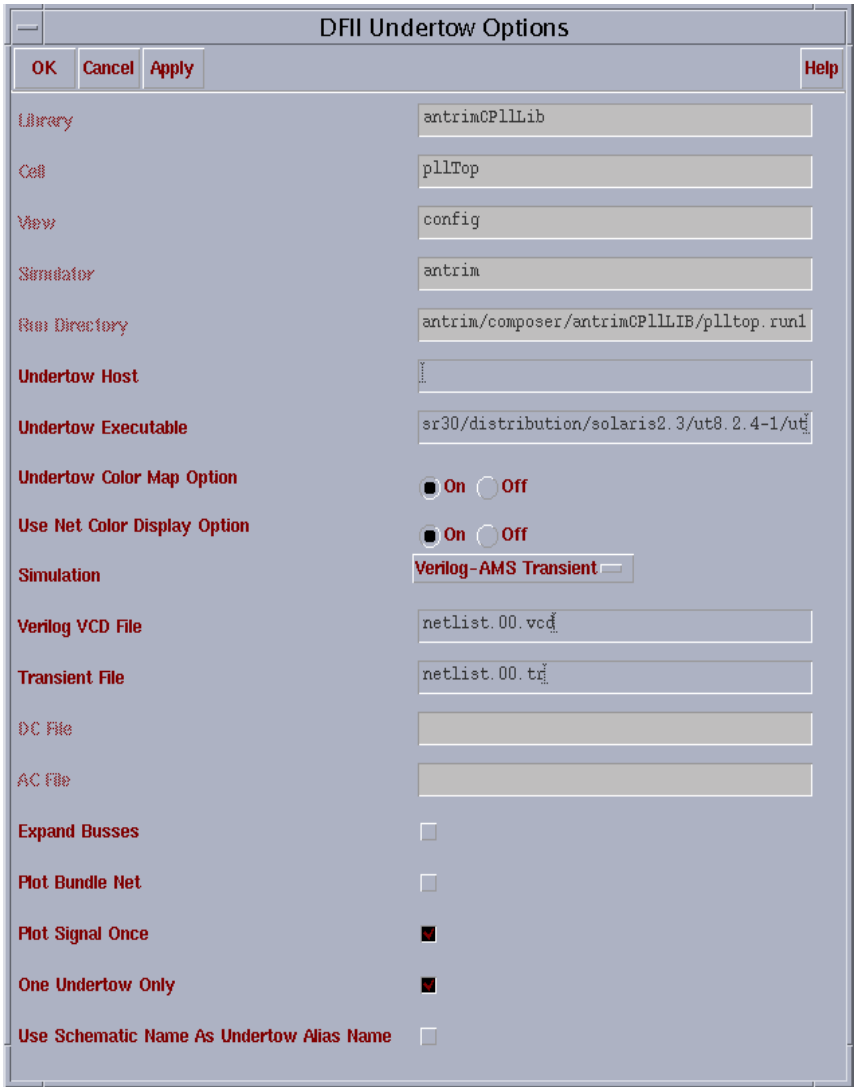
- **Start Undertow**

  This menu item invokes Undertow through a Skill IPC process. It also enables those probing fields.

- **Options...**

  This menu item pops up a form, in which you set options to customize the Undertow-dfII integration environment:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



- **Library**

  This field displays the library name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **Cell**

  This field displays the cell name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **View**

  This field displays the view name, captured from the specified **OSS** netlist directory.

  Note: This is a grey-out (inactive) field.

- **Run Directory**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This field displays the run directory name specified from the initialization.

Note: This is a grey-out (inactive) field.

- **Undertow Host**

  This field indicates the name of the host on which Undertow will run. By default, Undertow runs on your local host.

- **Undertow Executable**

  This field indicates the full path to the Undertow executable. To set this field, use the **UT_ROOT_DIR** UNIX environment variable.

- **Undertow Color Map Option**

  When you toggle this option to **On**, Undertow allocates resources to the color display.

- **Use Net Color Display Option**

  By default, this option is **On**, so it always uses the color of the probed net, as the displayed color of the waveform. If you set this option to **Off**, Undertow uses the signal's defined color to plot each signal.

- **Simulation**

  Use this pull-down menu to specify the types of simulation to run. Supported simulation types are:

  - Analog Transient
  - Analog AC
  - Analog DC
  - Verilog-AMS Transient
  - Verilog

  If you are running Antrim Analog DC simulation, select the **Analog DC** simulation mode.

  If you are running Antrim Analog AC simulation, select the **Analog AC** simulation mode.

  If you are running Antrim Analog Transient simulation, select the **Analog Transient** simulation mode.

- **Verilog VCD File**

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

This field specifies the name of a Verilog digital simulation file, such as a **vcd** file or a Veritools supported dump file. You must specify this field if you selected **Verilog-AMS Transient** or **Verilog** in the simulation field.

*Note: This field is greyed-out (inactive) if the simulation field is neither **Verilog-AMS Transient** nor **Verilog**.*

- **Transient File**

This field specifies the name of the analog Transient file. You must specify this field if you selected **Verilog-AMS Transient** or **Analog Transient** in the simulation field.

*Note: This field is greyed-out (inactive) if the simulation field is neither **Verilog-AMS Transient** nor **Analog Transient**.*

- **DC File**

This field specifies the name of the analog DC file. You must specify this field if you selected **Analog DC** in the simulation field.

*Note: This field is greyed-out (inactive) if you did not select **Analog DC** in the simulation field.*

- **AC File**

This field specifies the name of the analog AC file. You must specify this field if you selected **Analog AC** in the simulation field.

*Note: This field is greyed-out (inactive) if you did not select **Analog AC** in the simulation field.*

- **Expand Buses**

If this button is **ON**, and if you generated the netlist in a **Buses** netlist mode, then Undertow expands all bus signals that you selected from the schematic probing window, into individual bits, for display in Undertow.

- For example, the **data[2:3]** signal displays as data[2] and **data[3]** in Undertow, if the button is turned *On*.
- By default, the button is turned “Off”.

If you used the **Bit** netlist mode in the Verilog integration environment, to generate the netlist within the specified **OSS Simulation Run** directory, then this button has no effect.

- **Plot Bundle Net**

Use this button to plot a *bundle* signal as *bundle* waveform. By default, this option is set to **false**. If you turn this option **Off**, Undertow breaks the selected bundle net into separate pieces, such as buses or a scalar net, for display.

- For example, if the option is turned **On**, then the **a<2:44>,b,c** signal displays as **a<2:4>,b,c**.
- Otherwise, Undertow plots three signals, **a<2:4>**, **b**, and **c**.

- **Plot Signal Once**

  When you toggle this button **On**, Undertow does not plot any specified signal if the signal has already been plotted. By default, this option is set to **true**.

- **One Undertow Only**

  This button limits the number of Undertow executables that can run at any one time, to one. If you start a new Undertow session while another Undertow session is still running, this option kills the previous Undertow process.

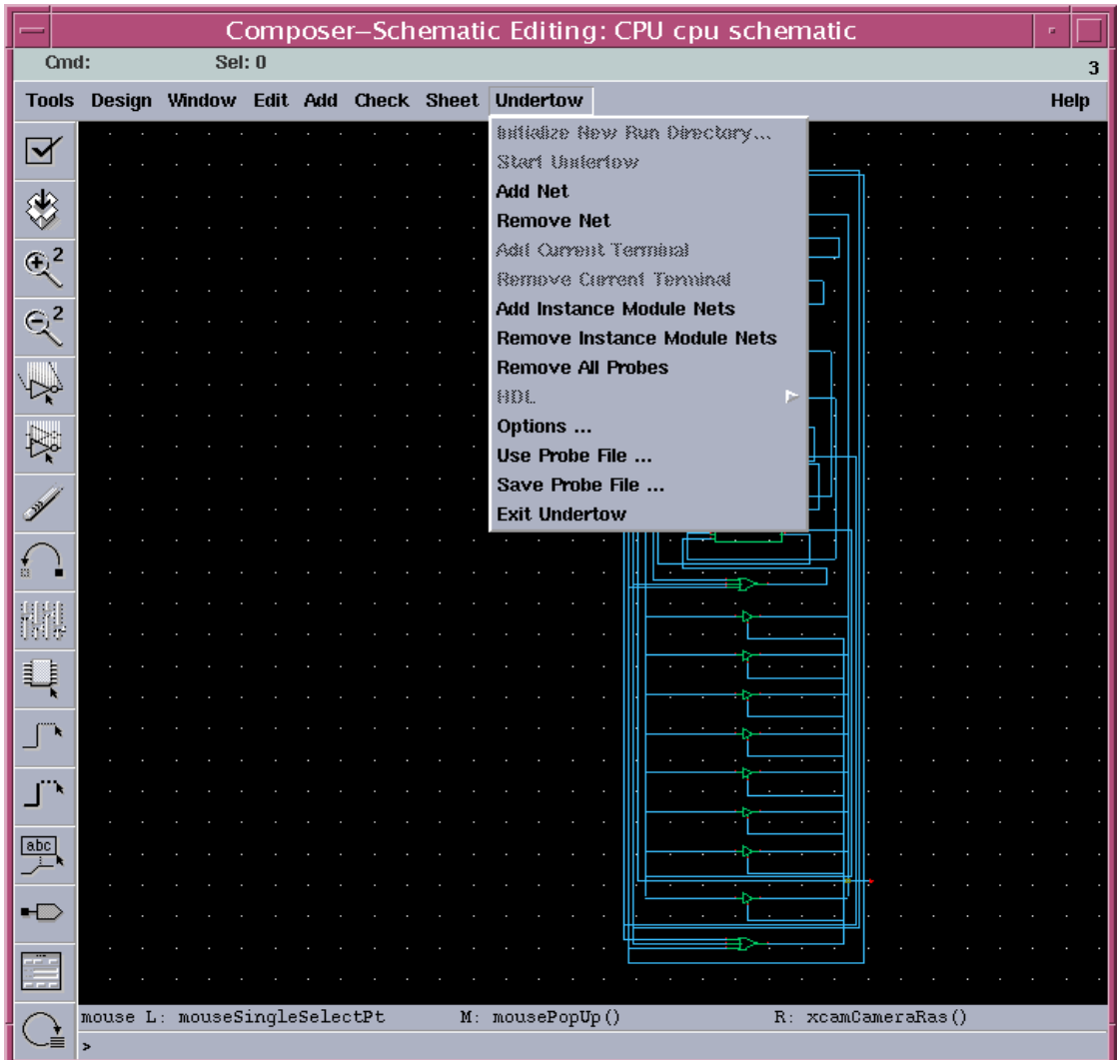
  The default is that you can run more than one Undertow process at one time, but all commands are always sent to the newest Undertow process.

- **Use Schematic Name As Undertow Alias Name**

  If this button is turned **On**, Undertow uses the schematic name space for display.

  For example, a schematic signal named **COUNTER/q<2>** maps to **test.top.COUNTER.q[2]** in a Verilog netlist. If the button is turned ON, then **COUNTER/q<2>** becomes an alias signal name for **test.top.COUNTER.q[2]**. The plot window in Undertow shows **COUNTER/q<2>**.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



After you select **Start Undertow**, the integration kit starts an Undertow **waveform** viewer through a Skill IPC process. When the Undertow **waveform** opens, some pull-down fields are enabled, so that you can start to probe signal waveforms from either a schematic or a text view.

- **Add Net**

  When you probe a net, this menu item lets you add the net to the plot in Undertow.

  After you select **Add Net**, you can use the schematic window to probe a net. If a net is properly probed, an **Add** signal message is sent, so that Undertow displays the probed net in the *waveform* window.

  This menu item is disabled if you use an HDL text viewing window.

- **Remove Net**

When you probe a net, this menu item lets you remove the net from the plot in Undertow.

After you select **Delete Net**, you can use the schematic window to probe a net. If a net is properly probed, a **Delete** signal message is sent, so that Undertow deletes the probed net from the *waveform* window.

This menu item is disabled if you use an HDL text viewing window.

- **Add Instance Module Nets**

  When you probe a net, this menu item lets you add referenced nets to the plot in Undertow.

  After you select **Add Instance Module Nets**, you can use the schematic window to probe a referenced net. If a net is properly probed, the Undertow *waveform* window displays all nets in the same object instance, that are referenced by the probed instance net.

  - This menu item is enabled if the simulator is **Verilog**.
  - This menu item is disabled if you use a Verilog HDL text viewing window.

- **Remove Instance Module Nets**

  When you probe a net, this menu item lets you remove referenced nets from the plot in Undertow.

  After you select **Remove Instance Module Nets**, you can use the schematic window to probe a referenced net. If a net is properly probed, the Undertow *waveform* window no longer displays any nets in the same object instance, that are referenced by the probed instance net.

  - This menu item is enabled if the simulator is **Verilog**.
  - This menu item is disabled if you use a Verilog HDL text viewing window.

- **Remove All Probes**

  Remove All Probes sends a message to Undertow, to remove all plotted signals from the schematic view. This options clears all probed nets and instances within the schematic.

  This menu item is disabled if you use a Verilog HDL text viewing window.

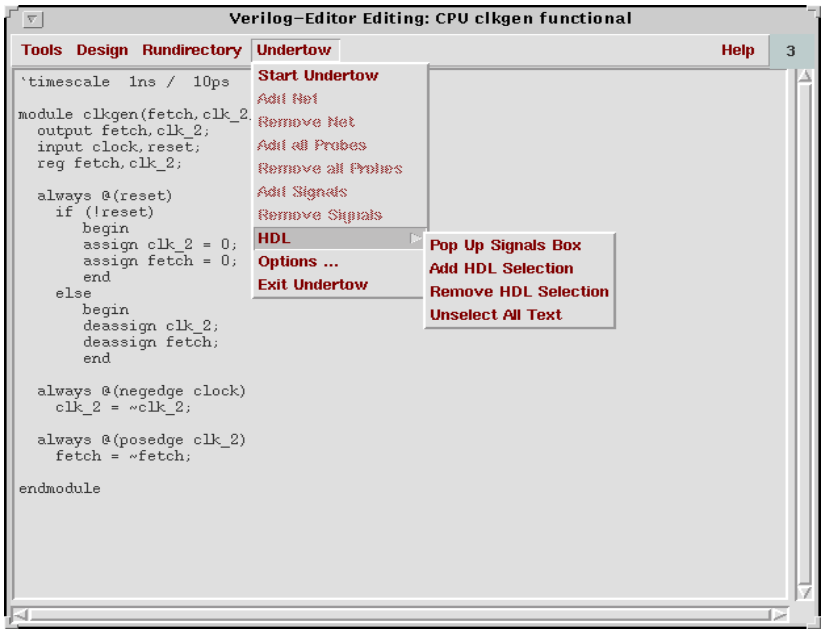
- **HDL->**

  This submenu contains commands to select signals in a HDL text viewing window, such as **verilog** or **antrim**.

  - This menu item is disabled if the opening window is a schematic window.

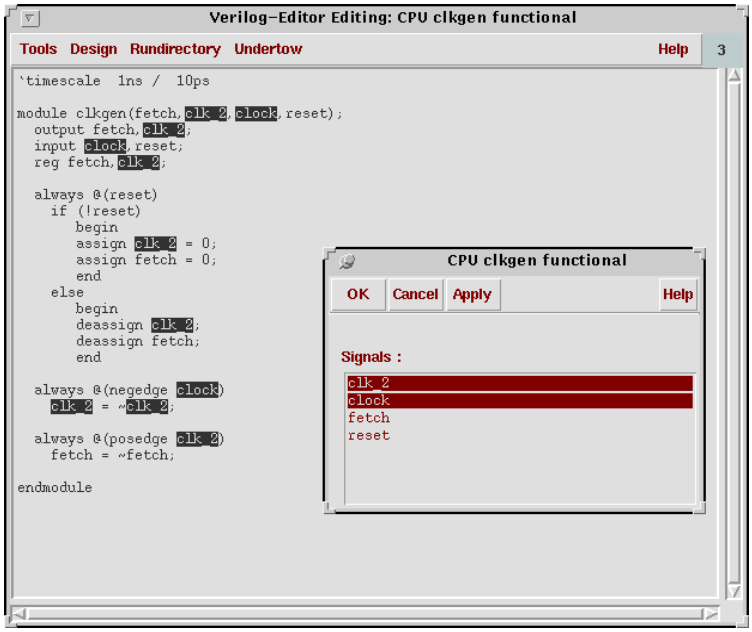
 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- This menu item is enabled if you use a HDL text viewing window.



- **Pop Up Signals Box...**

  This menu item pops up a signal box, to display all signals for the HDL *text viewing* window. If you select multiple signals from the signal box, the form callback automatically searches and highlights all matched signals, within the HDL *text viewing* window.



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Add HDL Selection**

  1. Either manually highlight a signal, or use the Text Signal Box option.
  2. Select **Add HDL Selection**.

     Undertow displays all selected text(s).

- **Remove HDL Selection**

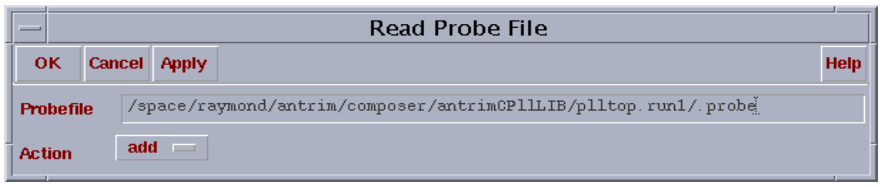
  1. Either manually highlight a signal, or use the Text Signal Box option.
  2. Select **Remove HDL Selection**.

     Undertow removes all selected text(s).

- **Unselect all Text**

  Clears all text signal selections.

- **Use Probe File...**



  You can specify a probe file to add or delete from the schematic view.

  If you select the **add** operation, the integration skill reads the probe file, to probe any missing net/instance to the schematic view. It also sends a message to Undertow, to plot the waveform.

  This menu item is disabled if you use an HDL text viewing window.

- **Save Probe File...**

  You can save the current probed instance and net, from the schematic view.

  This menu item is disabled if you use an HDL text viewing window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

- **Exit Undertow**

  This menu item exits Undertow.

Instance Hierarchy Traversal

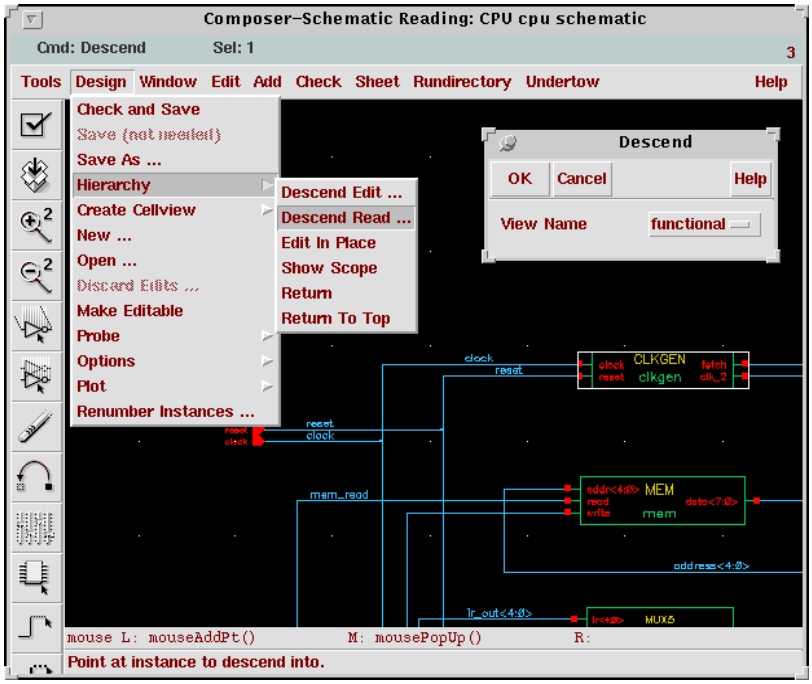
1. Open a schematic or Verilog-HDL text viewing window, from the topmost cell of your OSS simulation run directory.

   This step ensures that the OSS environment successfully determines the instance hierarchy.

2. To traverse the design hierarchy, select **Composer => Design => Hierarchy => descend Read**.

3. Specify the right switching view for a selecting instance.

   This step ensures that the OSS environment determines the instance hierarchy for OSS name mapping.



 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Using Undertow in the Cadence DFII Verilog Integration Environment

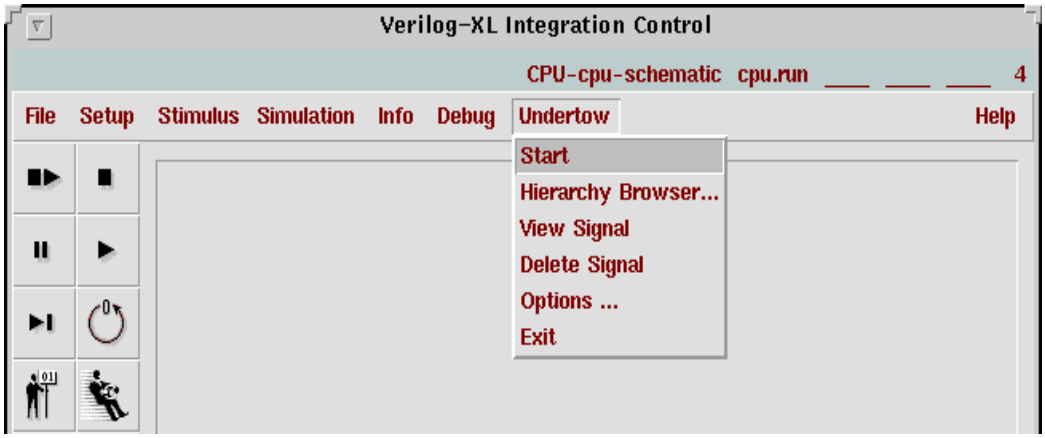
This section describes how to use Undertow to display signals within the Verilog Integration environment.

Usages

1. If you have not already done so:
   - Run the Verilog netlister, to generate the netlist, and
   - Run the simulation, to generate the Verilog VCD file.
2. Open a Verilog-XL **Integration Control Encapsulation** window.
3. Specify an **OSS Simulation Run** directory.
   - The **Encapsulation** window opens.
4. Make sure that:
   - The **Encapsulation** window shows simulation results, in Verilog VCD format, and
   - The netlists have been generated.

   *Note: This release supports post-simulation probing **only** for Undertow. If you run Verilog-XL simulation in interactive mode, within the Verilog Integration environment, Undertow is not supported.*
5. Verify that a new menu, named *Undertow*, appears in the banner menus of the Verilog-XL **Integration Control Encapsulation** window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



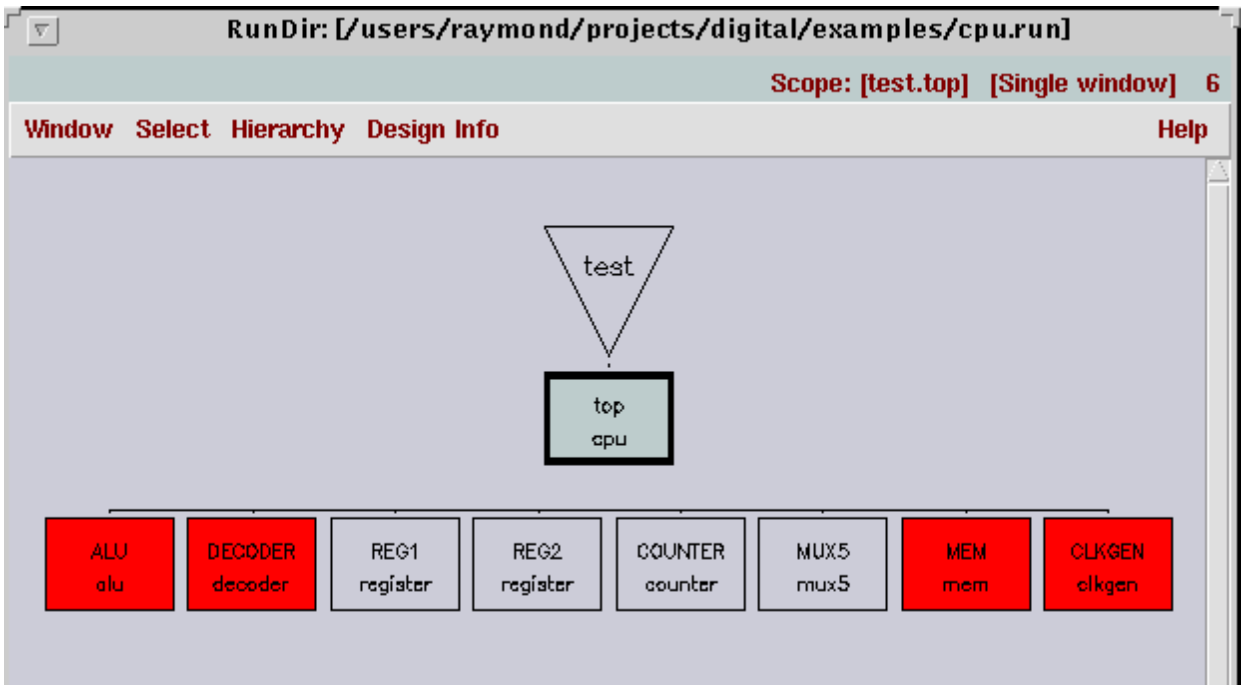
The Undertow menu contains the following menu items:

- **Start Undertow**

  This menu item invokes **Undertow**, through a Skill IPC process.

- **Hierarchy Browser...**

  This menu item pops up the **Hierarchy Browser** of the Verilog Integration.



- **View Signals**

  1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

2. Select signal(s), from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.

3. Select **View Signal**.

   The Undertow window displays the selected probed signals.

- **Delete Signals**

  1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

  2. Select signal(s), from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.

  3. Select **Delete Signal**.

     The Undertow window no longer displays the selected probed signals.

- **Options...**

  This menu item pops up the same options form described in **Standalone Usage** mode. Refer to Section , “Standalone Usage Of Undertow-Cadence DFII Integration”.

- **Exit**

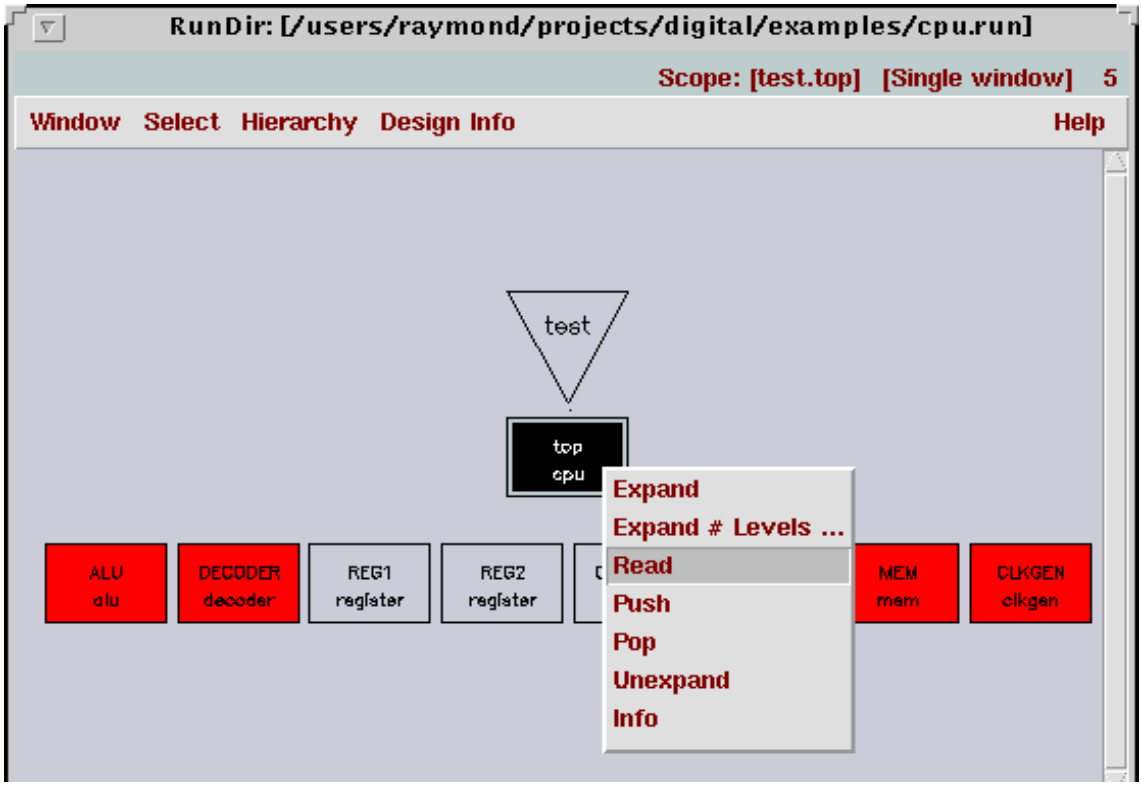
  This menu item exits Undertow.

Signal Probing In Verilog Integration Environment

1. From the **Hierarchy Browser**, open either the *schematic viewing* window, or the Verilog-XL-HDL *text viewing* window.

2. Select signal(s) to probe, from either a schematic viewing window, or the Verilog-XL-HDL text viewing window.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved



Preparing a Verilog VCD File For Undertow

Add the following lines into your Verilog stimulus file:

```
initial
  begin
    ....
    $dumpfile(“verilog.vcd”);
    $dumpvars;
    ......
  end
```

These lines enable Verilog-XL to generate simulation results in Verilog VCD format.

After the Verilog-XL simulation finishes, a simulation Verilog VCD file, named *verilog.vcd*, appears in the *run* directory.

Preparing Veritools dump file for Verilog-XL

Add the following lines into your Verilog stimulus file:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
initial
 begin
  ....
  $vtDumpvars;
  ......
 end
```

These lines enable Verilog-XL to generate simulation results in Veritools *dumpfile* format.

Note: Refer to , for more details about how to use the Veritools PLIs.

Preparation Of Verilog Netlists In Verilog Integration Environment

- ***OSS "si" mode***

  1. Set up an OSS initialized file named si.env
  2. Copy the Cadence cds.lib file into the simulation run directory.
  3. Do either of the following:
     - Set up a OSS standalone **SI** batch call, or
     - Invoke the **si** binary, then type **netlist()**.

- ***Verilog Integration Environment Interactive Mode***

  1. Open the Verilog-XL Integration **Encapsulation Control** window.
  2. To trigger the Verilog HNL netlister, either:
     - Press the **Start Interactive** icon from the **Encapsulation** window, or
     - Type **netlist()** in CIW, to generate a Verilog netlist for the design.

Preparing Netlists In Antrim Integration Environment

Refer to Antrim's manual for how to prepare netlists for the Antrim simulator.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Interface With Undertow From Other Applications

This section describes how to interface other applications with Undertow.

Invoking Undertow From Other C Applications:

Start Undertow with command line arguments:

1. Use the following command line argument to start Undertow:

   "ut -v filename.vcd -P 0/1" for stdin/stdout

2. Type commands into the same shell terminal that you used to start Undertow.

   Undertow reads commands from the standard input:

3. In the same terminal, type:

   ADD test.top.reset

   ADD test.top.COUNTER.q COUNTER/q

   Undertow displays **test.top.reset** and **COUNTER/q** in the plot window.

4. Use the same mechanism, *cat* a file into **ut** as input:

   "cat commands.txt | ut -v filename.vcd -P 0/1"

   where the commands.txt file stores all of the Undertow-recognized commands, line by line.

Set up a link between Undertow and Application by a C

routine:

A parent program (your **C** application) can set-up pipes, for input/output with Undertow. It can then *fork/exec*, with the command line arguments, for the *fileno()* of the Unix pipes:

fork() => exec()

with

"ut -v filename.vcd -P fileno(in_pipe)/fileno(out_pipe)”

0.0.0.1 C Example:

```
void InvokeUndertow()
{
 char *argv[4];
 char *command;
 int in_fds[2]; /* incoming file descriptor */
 int out_fds[2]; /* outgoing file descriptor */
 int pid;
 strcpy(buf0, "ut");
 argv[0] = buf0;
 strcpy(buf1, "-p");
 argv[1] = buf1;
 sprintf(buf2, "%d/%d", out_fds[0], in_fds[1]);
 argv[2] = buf2;
 argv[3] = NULL;
 envp[0] = NULL;
 if (!(pid = fork())) {
 close(in_fds[0]);
 close(out_fds[1]);
 /* command - full path for ut executable */
 execve(command, argv, environ);
 perror(argv[0]);
 exit(-1);
 }
 close(in_fds[1]);
 close(out_fds[0]);
 /* in_fds[0], out_fds[1], pid where
 pid - will be the process id for future communication
 in_fsd - file descriptor the UT will write it to
 out_fds - file descriptor the application will write the
command to UT
 */
 /* Example to write command to UT :
 write( out_fds[1], &string[index], len);
 where :
out_fds[1] - write file descriptor to UT
string - command to send it to UT such as "ADD test.top.reset"
```

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

```
len - length of the String command
*/
}
```

Current Supported Undertow Commands

- Each command terminates with a newline (\n).
- Any other white space separates the arguments for a command.
- Currently, Undertow reads *only* the in_file_desc.
- Undertow does not write anything to the out_file_desc.

You can use the in_file_desc to send the following commands to Undertow:

ADD <options> <signal_name> [<alias_signal_name>

This command displays the specified signal, and assigns the specified name alias to the signal.

For example:

"ADD test.top.reset"

"ADD test.top.COUNTER.clear"

If you specify an <*alias_signal_name*>, the signal displays as the <*alias_signal_name*>.

For example:

"ADD test.top.reset reset"

"ADD test.top.COUNTER.clear COUNTER/clear"

ADD-ALL <options> module_name

This command displays all signals in the *module_name* module.

For example:

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

"ADD test.top.CLKGEN"

where *CLKGEN* is a module.

ADD-TO-OVERLAY <-color color_name> signal_name

This command uses the same arguments as the **ADD** command, but adds the signal to the selected overlay, instead of displaying it separately.

For example, adding a new overlay:

ADD -select -color Red top.a1

ADD-TO-OVERLAY -color Green top.b2

ADD-TO-OVERLAY -color Blue top.c3

For example, adding to a selected overlay:

ADD-TO-OVERLAY -color Red top.a1

ADD-TO-OVERLAY -color Green top.b2

ADD-TO-OVERLAY -color Blue top.c3

BUNDLE <options> bundle_name <signal_name> <signal_name> ...

This command creates and displays the *bundle_name* bundle, composed of the specified signals. If you do not specify any signal names, the bundle is deleted.

For example:

“BUNDLE my_bundle1 test.top.ir[5:7] test.top.dec[2:1]"

DELETE <signal_name>

This command removes all signals with the specified name, from the display Undertow window.

For example:

"DELETE test.top.reset"

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

"DELETE test.top.COUNTER.clear"

GET-COLOR <signal_name>

This command reports the color of the *signal_name* signal. The return format is:

"COLOR *signal_name color_name*"

For example:

"GET-COLOR test.top.reset"

Undertow returns:

"COLOR test.top.reset RED"

<options>

-color <color_name>

This option performs the selected operation, using the specified color.

*Note: Undertow currently does not support different colors for logic waveforms. You can assign different colors **only** to analog waveforms.*

-select

This option toggles the signal name on (selected), when you add the signal. Use this command option with the **ADD-TO-OVERLAY** command.

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

 Proprietary and Confidential Copyright Veritools 2004 All Rights Reserved

Index

Numerics
3DB.script file 391
3X Current Time interval 118
5X Current Time interval 118
A
absolute values 229
AC analysis 221
AC file 523
ACOS 188
ACOSH 188
ACOSPI 188
Actions Icon
  Source Code Window 237
Actions icon 67
Actions menu 295, 311
activating
  windows 208, 383
Active View
  setting 350
Add Color To List field 97, 148
ADD command 535, 537
Add File To List button 249, 251, 268, 287, 300, 302
Add HDL Selection command 528
Add Instance Module Nets command 526
Add Net command 525
Add New Pane command 122, 215
Add New Pane, Arbitrary X command 215
Add New Pane, Time command 215
Add Operator to Expression 316
Add Pane Horizontal 267
Add Pane Vertical 267
Add Pane, Horizontal command 349, 350
Add Pane, Vertical command 349, 350
ADD signal 184
Add to Open Files command 70
ADD-ALL command 537
adding
  arrows 217
  breakpoints 314
  labels 206, 213, 382, 388
  list-box 207, 382
  panes 46, 122
  Perl code 203, 378
  text 217
  value areas 123
  widgets 212, 388
Adding and Removing Value Area command 123
adding panes
  State window 267
ADD-TO-OVERLAY command 538, 539
Alias By Name command 84
Alias By Value command 86
Alias field 85
Alias files
  filtering 87
Alias Modify dialog 81, 85
Alias Names 86
Alias New dialog 81, 85
aliases
  GoTo tool 50
  signals 48, 50
Aliases Displayed dialog 50
All button 126
All Icon
  Source Code Window 237
All icon 67, 295
all signals
  display 39
All Time intervals 118
All/First button 130
Alter Signal dialog 320
Alter Signal window 313, 314
Amplitude.script file 391
Anagram 15, 31, 51
analog
  features 140
  functions 134, 176
  tools 166
  waveforms 220
Analog AC
  simulation 522
Analog Analyzer 132, 167
Analog Color command 77, 216
Analog command 77
Analog Configuration button 145, 147
Analog Configuration icon 62, 183
Analog Configure dialog 148
Analog DC
  simulation 522
Analog Expression field 190
Analog Expression window 179, 191, 192
analog expressions 178

analog extensions
Undertow 390
Analog Features command 62
Analog Filter Builder window 178
analog filters 192
Analog Function Calculator 186
Analog Functions 177
Analog Functions command 186, 228
Analog Functions menu 220
Analog Functions window 178, 181, 187
Analog Functions, Edit Selected command 181, 220
Analog Functions, Selected button 135
Analog Grids button 100, 151
analog icons 141
analog signals
comparing 179
configuring 96, 147
converting to digital 169
measuring 208, 384
Analog To Digital window 170
Analog Tools Only (No Digital) button 98, 149
Analog Transient
simulation 522
analog waveforms
comparing 228
comparing to digital waveforms 229
converting to digital waveforms 229
properties 181
Analog, Points command 77
analog_scripts.tar file 391
analysis
AC 221
fall time 173
pulse width 175
rise time 172
toggle 31, 51
analyzing
coincidence 352
AND operator 364
Annotate 110
Annotate command 313
Annotate Hexadecimal/Binary 318
Annotate Waveform command 110
annotating
binary 318
hexadecimal 318
waveforms 110
Antrim 515
netlist 534
simulator 522
antrim window 526
aparmeters
pass_limit1 194
append command text 308
Apply button
configuring 201, 376
arbitrary expressions
results 230
Arbitrary X command 215
Arbitrary X pane 46, 180, 192, 195
arrays 363
arrows
adding 217
ASCII command 76
ASIN 188
ASINH 188
ASINPI 188
Assign Scales field 183
assignments
signals 269
associative arrays 363
ATAN 188
ATANH 188
ATANPI 188
attaching
scripts to buttons 213, 389
attributes
Browse 304
Simulator Executable 304
Simulator Type 304
AtX.script file 391
Auto Color Waveforms command 168
Auto Fit To Pane button 98, 149
Auto Scroll command 115
Auto Update command 118
Auto Update Off icon 118
auto-decrement operator 365
auto-increment operator 365
Automatic printing 74
AVERAGE 188
Average function 392
Average.script file 391

averaging
waveforms 133
axis
linear 215
log 215
offsets 215
plotting 215
axis labels 217
B
Back command 336
Back Icon
Source Code Window 236
Back icon 296
back icon (source code viewer) 296
backward
stepping 267
base file 277
Baseline.script file 391
batch mode 280
batch operation
Undertow suite 248
batch simulation 252, 283
Begin Search button 296
BESSEL 188
binary
annotating 318
Binary command 76
bindkeys 218
bit netlist 518
Black & White Only field 75
blocks
commands 363
Boolean functions
signals 128
Boolean Functions window 134, 176, 181
Border field 75
boundary
measurements 210, 385
box zoom 233
branches
VCT 259
Break Expression 317
Break Point window 314
Break Points 316
Break Points dialog 315
Breakpoint dialog 319
Breakpoint On Line icon 355
Breakpoint On Signal icon 355
Breakpoint window 315
breakpoints 300
adding 314
deleting 314, 324
disabling 324
loading 314
modifying 314
removing 324
saving 314
setting 316, 324
source code 258
Source window 323
BreakPoints command 314
Breakpoints command 300, 302
Browse attribute 304
browser
Source Code Browser 236
browsers
Gate 295, 327
Module 328
module 36
modules 262
Source Code 260, 286, 322
Source Code Module 271
State 344, 351
State Machine 343, 348
bubbles
state 275
Builder 201, 376
starting 206, 381
Builder window 197, 198, 201, 208, 373, 377, 383, 384
Menubar 211, 386
templates 212, 387, 388
Toolbar 211, 387
BUNDLE command 538
Bundle Connections command 336
Bundle Name field 47
Bundle New command 47
Bundle Signals New command 94
Bundle window 94
bundled signals 48
bundling
signals 81
waveforms 83
bundling signals 46
buses 46, 63, 81

aliases 48, 50
expanding 80
buses netlist 518
BUTTERWORTH 188, 193
button definitions 307
button label 308
buttons 300
Add File To List 249, 251, 268, 287, 300, 302
All 126
All/First 130
Analog Configuration 145, 147
Analog Functions, Selected 135
Analog Grids 100, 151
Analog Tools Only (No Digital) 98, 149
Apply 201, 376
attaching scripts 213, 389
Auto Fit To Pane 98, 149
Begin Search 296
Clear 87
Clear All Groups 64
Clear Found 130
Clear Last Group 63
Command Line Interface 98, 149
Copy 87
Create and Load Fast File 136
Cut 87
debugging 354
Delete 64
Display 64, 296
display 39
Display, Add Overlay 146
Display, Add Plot 142, 146
Display, New Overlay 146
Expand Buses 523
Explode Bus 80
File Group 64
Find All 130
Find List 130
Find Selected 130
Fit To Pane 143
Fit to Pane 219
Go Back 45, 61
Go To 61
GoTo 45
Group 218
Group Definition 64
Help 64
Highlight Found 130
Highlight Net 263
Ignore Initial 'x' 130
List 126
List of signals to ignore 130
Lock X At First Point 104, 155
Lock X At Value 106, 157
Lock Y At First Point 105, 156
Lock Y At Value 107, 158
Measure Tag 145
Measures 124, 142
Modify 64
Module 280
Modules 269, 325
Monitors and Dumps 488
New Group 64
No Tag Lock 104, 155
One 126
One Undertow Only 524
Paste 87
Paste, Add To Overlay 141
Paste, New Overlay 141
Plot Bundle Net 523
Plot Signal Once 524
Save Groups 65
Schematic Tool 62
Select All 79, 87
Selected 126
Set Master Signal 142
Set Scope 79
Set Selected 122
Show Drivers 257
Snap to Source Code 271
State Tool 62
Title Tag 143
Unselect all Text 528
Use Analog Line Patterns 98, 149
Use Glyphs 337
Use HSpice Nomenclature 98, 149
Use Project Name for Files 488
Use Schematic Name As Undertow Alias Name 524
Use Text 337
Value 123
Value/Glitch 130
Values Tag 144
Variable 269, 325
Variables 280
zoom 40
Zoom In, Both 141
Zoom In, Vertical 141
Zoom Out, Both 141

Zoom Out, Vertical 141
Buttons command 300, 302, 309
Buttons Definition window 307
Buttons widget 198, 374
C
C code 536
Cadence Composer 517
Cadence dfII 515
Calculate Average command 133, 170
Calculate Slope command 133, 162, 170
calculating
waveform averages 133
waveform slopes 133
CBRT 188
cds.lib file 534
CDS_INST_DIR environment variable 468
cdsinit file 516, 517
Cell library 521
ChapterTitle
CHAPTER 11 Undertow Inegration Guide with Thrid-party Tools 515
CHEBYSHEV1 188, 193
CHEBYSHEV2 188, 193
chmod command 20
Choose Icon
Waveform Window 240
Choose icon 58
Choose window 272, 277
chooser
icons 146
Chooser icon 66
Chooser toolbar
icons 66
Chooser window 275, 489
choosing
state diagrams 274
Clear All Groups button 64
Clear button 87
Clear command 335, 347
Clear Cursors command 108
Clear Errors menu 126
Clear Found button 130
Clear Last Group button 63
clearing
cursors 108
cleartrace command 486
click
double 233
right 233
single 233
clock signals 238
CLOSE 184
Close All Files command 70
Close command 332, 346
Close Files command 70
close interactive tool 299
Close Interactive Tool command 299
CloseAllFilesAndOpen function 367
closing
files 70
code
adding 203, 378
COEFPS 188
coincidence
analyzing 352
Coincidence icon 352
coincident states
showing 275
Color field 182
colors 216
signals 539
Undertow 522
command blocks 363
command line
examples 247
options 247
starting Undertow 33
Command Line Interface 183, 184
Command Line Interface button 98, 149
command status 259
commands
ADD 184, 535, 537
Add HDL Selection 528
Add Instance Module Nets 526
Add Net 525
Add New Pane 122, 215
Add New Pane, Arbitrary X 215
Add New Pane, Time 215
Add Pane, Horizontal 349, 350
Add Pane, Vertical 349, 350
Add to Open Files 70
ADD-ALL 537
Adding and Removing Value Area 123
ADD-TO-OVERLAY 538, 539
Alias By Name 84

Alias By Value 86
Analog 77
Analog Color 77, 216
Analog Features 62
Analog Functions 186, 228
Analog Functions, Edit Selected 181, 220
Analog, Points 77
Annotate 313
Annotate Waveform 110
Arbitrary X 215
ASCII 76
Auto Color Waveforms 168
Auto Scroll 115
Auto Update 118
Back 336
Binary 76
BreakPoints 314
Breakpoints 300, 302
BUNDLE 538
Bundle Connections 336
Bundle New 47
Bundle Signals New 94
Buttons 300, 302, 309
Calculate Average 133, 170
Calculate Slope 133, 162, 170
chmod 20
Clear 335, 347
Clear Cursors 108
cleartrace 486
CLOSE 184
Close 332, 346
Close All Files 70
Close Files 70
Close Interactive Tool 299
Configuration 167, 217
Configure 66, 79, 87, 208, 337, 349, 383
Connectivity File 333
Continue 310
Convert To Digital 169
Convert to Digital 229
Copy 335, 347
Create Auxiliary File 76
Create Eye Diagram 133, 171
Create Fast File 76
Create Reduced File 76
Cursors 42, 108
Cut 335, 347
Decimal 76
DEL ALL 184
DEL X 184
DEL Y 184
DELETE 538
Delete Signals 532
Details Window 313
Display 336
Display Current Drivers 336
Display Drivers, Selected Signal 128
Down 336
Dump Bundles 95
Dump Names 95
Dump Signals 95, 219, 230
Dump Value Aliases 95
dump.fast 139
dumpfile 255, 490
dumpvars 119, 490
Edit Source File 312
End Time 131
EXIT 185
Exit 532
Exit Undertow 529
Expand All Drivers 336
Expand All Loads 336
Fast Selection 55
File Suffix 87, 178
Find 312
Find All 132
Find List 132
Find Selected 132
finvc 507
Fit To Pane 113
Fix Cursor 55, 108
Floating Choose Window 58, 124
Floating Group Window 124
Floating Icon Bar 124
Floating Menu Bar 124
Floating Windows 52, 63
Floating windows 53, 59
Fonts 54, 92, 217
Forward 336
FREE 185
Get Information 93
Get Information Dump Signals 94, 95
GET-COLOR 539
Go To 229
Go To Cursor 108
Go to State 348
GoTo 44
GoTo Error 61

GoTo LE 61
GoTo Marker 61
GoTo TE 61
GoTo Time 61
GoTo Transition 61
GoTo Value 61
Grid Signal 108
Grid Signal LE 108
Grid Signal TE 108
Grids and Scale 183
Grids and Scales 217
HDLLINT Report 323
HELP 185
Help 178
Hexadecimal 76
Highlight 207, 383
Highlight Errors 126
Highlight Net 336
History 310
Horizontal and Vertical Offset 183
Interpolate Current 168
Interpolate Voltage 168
Interrupt 310
ivsimcmp 255
Kill 310
Link Cursors 109
LIST 185
LIst 81, 85
List (expressions) 128
List of Signals to Ignore 132
LMACRO 185
lmgrd 19, 21
lmstat 19, 21
Load Additional File 275, 489
Load Batch Run 69, 312, 488
Load Step File 348
Lock Cursors 109
LOFF 185
Logic 77
MACRO 185
Measure Tags 113
Menubar 210, 386
Modify Selected 81, 85
Modify Selected (expression) 128
Modify Selected Signal 313
Monitors and Dumps 488
Name Alias 81, 85
Names 336
Navigator 107, 313
New (expression 128
NEW signal 185
New Source Code 323
New Waveform 323
Normal 76
Octal 76
Ones Compliment 76
OPEN 185
Open 69, 70, 299, 332, 346, 517
Open Additional File 36, 68
Open Waveform 69
Open Window 314, 318
Open/Close 135
Open/Close File 68, 219, 249, 252, 287
OpenAdditonalFile 276
OpenFiles 276
Over Draw Mode 348
Overlay Files 92
Overlay Waveforms 163, 168
Pane Zoom Analog 160, 168
Paste 335, 347
PAUSE 185
Persistence 348
PLI static 453
Pop Up Signals 527
PRINT 185
Print 332, 346, 362
Print Signals 230
Print To File 334, 347
Print To Printer 334, 347
Projects 300, 302
Properties 91
QUIT 185
Quit 299
Real Time 118
Remove All Probes 526
Remove HDL Selection 528
Remove Instance Module Nets 526
Remove Net 525
Remove Pane 123, 349
Remove Tag 112
Reread Dumpfiles 215
Reset 310
Restore 65, 71, 230
Run 310, 464, 488
RUN filename 185
Run HDLLINT 312
Save 65, 71, 218, 299, 302, 332, 346
Save All 71, 72

Save As 204, 205, 214, 299, 332, 338, 346, 379, 380, 389
Save File 333
Save Groups 65, 71
Save Probe File 528
Schematic 261, 271, 272, 323, 328, 341
Schematic Browser 128
Script 200, 376
Script tool 205, 381
Script Type 204, 205, 214, 379, 380, 389
Scripts 201, 210, 377, 385
Scroll 90% 117
Scroll LE 115
Scroll Percent 116
Scroll Set Interval 117
Scroll TE 116
Scroll To Max Time 116
Scroll To Signal 57, 117
Select All 335, 347
Selected Analog Waveform 180, 192, 195
Selected Analog Waveform Properties 161, 181, 215, 216
Set Breakpoint On Selected Signal 313
Set Default 122
Set Display Interval 117
Set Tag Text 112
settrace 486
Show Connectivity 128
Show Driver 309
Show Drivers 336
Show Hierarchal Names 128
Show Histogram 167
Show Loads 309, 336
Show Selector 206, 337, 349, 381
Show Whole Schematic 336
SIGNALS 185
sigt 211, 387
sigthresh 212, 387
Simulate for a Time 355
Simulator Options 250
Simulator Type 250
Single Scope Only 336
Snap/Source 261
Source Code 248, 251, 268, 286, 300
Source Code Files/Simulation Files 463
Source Window 318
Start Interactive Tool 36
Start New Undertow 72, 218
Start Time 131
Start Undertow 525, 531
State 274
State and Schematic 274
State Coincidence, Display 348
State Coincidence, Step 348
State Coverage 348
State Diagram 323
State Sequence 348
State Table 348
States 266
States and Schematic 261, 266, 271
States and Schematics 328
Step Back 348
Step Forward 348
Stop Layout 336
Sync 218
Sync Panes 122, 218
Sync Undertows 73
Tag Anchors 112
Tags 43, 110, 217
tar 19
tar -xvf 391
Target 109
Template 200, 376
Text (Choose) 78
Tile Window 314, 322
Tile Windows 261, 266
Time Scale 120
Title Tag 217
Title Tags 111
Toolbar 211, 386
Trace 300, 323
trace 255
Twos Compliment 76
uncompress 19
Undertow 518, 537
Up 336
Update 312, 336
Upscope 336
Use Probe File 528
ut 34, 246
ut -iv 248, 252, 253, 255, 284, 285, 286, 289, 292, 293, 456, 457, 458, 463, 469, 492
ut -v 465, 469, 535, 536
utConnectivity 488
utLimitVCD 139, 490
Value Tags 112
vcd 255
vcsi 457

VCT Report 300
View Signal 532
VTCOMPRESS 255, 455, 456, 457, 463, 465
vtDump 249, 490
vtDumpMem 489
vtDumpMemAddr 489
vtDumpvars 249, 255
vtrun -trace 455, 486
vtrun -tracefile 455, 486
vtTrace 486
vtTracename 487
Wave-Display 318
Waveform 300
Waveform Height 41, 160
Waveform Heights 217
Working Dir 87
Working Directory 178
WRITE 185
X 186
XLIN 186
XLOG 186
XVAR 186
Y 186
YLIN 186
YLOG 186
YRANGE 186
Zoom 1 to 1 348
Zoom Full 336, 348
Zoom In 336, 348
Zoom Out 336, 348
Zoom Time 168
comments 364
Perl 362
CompareAndInsertInHash function 397
CompareLogicSignals function 367
comparing
analog signals 179
analog to digital 229
analog waveforms 228
crossing points 229
signals 127
comparing files 31, 51
comparison operator 365, 366
compiled simulators 250, 254, 285
compiling
MTI 465
PLI 463
complete templates 213, 389
COMPLEX 189
Composer 517
compression modes 483
concatenation assignment operator 365
concatenation operator 365
Configuration command 167, 217
Configure command 66, 79, 87, 208, 337, 349, 383
Configure dialog 337
configuring
analog signals 96, 147
Apply button 201, 376
Value window 87
widgets 199, 375
Connection Analyzer 488
connections
finding 263
connectivity
netlists 51
Connectivity Analyzer 127
connectivity file - create 305
Connectivity File command 333
Connectivity File field 305
Connectivity History window 128
contents
memory 275
Continue command 310
continue statement 364
Continuous 316
controlling
panes 90
zoom 90
Convert To Digital command 169
Convert to Digital command 229
converting
analog to digital 169, 229
Copy button 87
Copy command 335, 347
Copy Icon
Waveform Window 240
Copy icon 54, 56, 67, 295
COS 189
COSH 189
cr_inca program 469
cr_inca script 468
Create and Load Fast File button 136
Create Auxiliary File command 76
Create Connectivity File field 305

Create Eye Diagram command 133, 171
Create Fast File command 76
Create Reduced File command 76
CreateAnalogSignal function 367
CreateNewVariable function 367
CreateXAxis function 367
creating
  Fast Files 69, 135
  files 76
  labels 202, 377
  Memory Dump file 489
  scripts 201, 377
  templates 204, 210, 379, 385
  trace dump file 486
  Verilog executable 453
  waveforms 219
  windows 201, 208, 377, 383
crossing points
  comparing 229
cshrc file 516
Ctrl signals 46
current
  interpolating 132
Current Display Time interval 118
Current pane 90, 160
Current Value widget 208, 383
cursors
  clearing 108
  linking 109
  locking 108, 109
  moving 108, 124
  placing 124
  setting 108
  T0 61, 159, 209, 384, 391
  T1 159, 209, 384, 391
  T2 159
  targets 109
  time 42, 55
  time values 53
Cursors command 42, 108
Cursors Significant Digits field 99, 150
Cut button 87
Cut command 335, 347
Cut Icon
  Waveform Window 240
Cut icon 54, 56
CutSignals function 367
D
DAEMON lines
  licenses 20
data
  loading 117
  signals 220
DB10 189
DB20 189
DC file 523
debugging buttons 354
Decimal command 76
def file 348
DEL ALL 184
DEL X 184
DEL Y 184
Delete button 64
DELETE command 538
Delete Signals command 532
deleting
  breakpoints 314, 324
DeltaX.script file 391
DeltaY.script file 391
demo licenses
  setup 19
descending
  hierarchy 529
DeselectAllSignals function 367
design - importing 300
design files window 464
destroying
  widgets 200, 375
Detail window 312
Details menu 318
Details Window command 313
DFII
  Verilog 530
Dialog 49
dialog
  Aliases Displayed 50
  GoTo 44
  new alias 49
  New Bundle 47
  Print 75
  print 74
dialog boxes
  File 69
dialogs

Alias Modify 81, 85
Alias New 81, 85
Alter Signal 320
Analog Configure 148
Break Points 315
Breakpoint 319
Configure 337
File Selection 64
Files 301
Find Logic 129
Fonts 93
Inspect Signal 321
Open Dump File 135
Open File 35
Open Source 301
Output 94
Overlay 92
Properties for Signal 182
Pulse Width Analyzer 176
Results 318
Search 319, 320
Sync Options 305
Test Analyzer 125
Time Scale 121
Trace Filter 319
Trigger Conditions 131
Value Tag 113
Variable 213, 388
WaveForm Heights 122
DIFF 189
digital signals 169
digital waveforms
comparing to analog waveforms 229
directories
OSS Simulation Run 518, 530
PLI 457
pli_example_files 139
Run 518
run 196, 361
scripts 392
signals 35
undertow 516
UT_ROOT_DIR 246
Directories panel 300
Disable 316
disabling
breakpoints 324
disk space 18
display
all signals 39
intervals 117
Display button 64, 296
display button 39
Display command 336
Display Current Drivers command 336
Display Current Drivers Icon
Source Code Window 236
Display Current Drivers icon 296, 355
display current drivers icon (source code viewer) 296
Display Drivers, Selected Signal command 128
Display Icon
Source Code Window 237
Display icon 58, 67, 295
Display window 52, 55, 127
Display, Add Overlay button 146
Display, Add Plot button 142, 146
Display, New Overlay button 146
displaying
drivers 257, 258
loads 257, 258
signal 326
signal colors 216
signals 273, 342, 517
source code 325
state table 275
waveforms 39, 55, 260
displaying signals
source code 270
displaying source code
waveforms 270
displaying wave forms (source code viewer) 300
displaying waveforms 31
DisplayVariableAddToOverlay function 368
divisions
pixels 101, 152
double click 233
Down command 336
Down Icon
Schematic Window 246
Source Code Window 236, 246
Down icon 297, 330
drag and drop 39, 233, 260, 269, 275, 325
drivers
displaying 257, 258

Dump Bundles command 95
dump file 139, 533
dump files 484
  generating 138
  Verilog 138
Dump Names command 95
Dump Signals command 95, 219, 230
dump trace file (source code viewer) 305
Dump Trace Info field 305
Dump Value Aliases command 95
dump.fast command 139
dumpfile 138
dumpfile command 255, 490
dumpFileName field 293
dumpfiles 215
  reading 70
dumpTraceInfo field 293
dumpvars 138
dumpvars command 119, 490
dumpvars program 119, 138, 139, 456, 468, 482
DutyCycle.script file 391
dynamic mode
  PLI 498
E
edges
  falling 173
  rising 172
  signals 44
Edit menu 125, 134, 178, 335, 347
Edit Source File command 312
editing
  expressions 220
  waveforms 219
EDITOR 22
EDITOR environment variable 312
editors
  Emacs 19
  vi 19
elements
  schematic 273
ELLIPTIC 189, 193
ELLIPTIC filter 195
Emacs editor 19, 22
Enable 316
Enable Results window 261
Encapsulated PostScript (see EPS)
Encapsulation Control window 534
Encapsulation window 517, 530
End interval 118
End Time 133, 174, 175
End Time command 131
End Time field 75, 172
EndTime function 368
Entry widget 198, 203, 374, 378
environment variables
  CDS_INST_DIR 468
  EDITOR 22, 312
  FINUSERPLIOBJ 507
  licenses 21
  LM_LICENSE_FILE 19, 20, 21
  LPR_CALL 75
  ModelTech 464
  PATH 22, 490
  path 456, 468
  PERLLIB 22
  PLI_BUILD 468
  PLIOBJS 463, 464, 465, 504
  UT_ROOT_DIR 21, 137, 456, 464, 468
  UT_WORK_DIR 22, 302
  VCSI_HOME 456
  ver_installdir 497
  VERITOLLS_LICENSE_FILE 22
  VERITOOLS_LICENSE_FILE 19, 20, 21
EPS files 229
equal operator 364
ERF 189
ERFC 189
Error icon 127
Error window 372
Escape Characters 9
events
  sequences 132
examples
  command line 247
  PLI 484
execute immediately 308
Execution Marks window 324
execution status 259
exercised states
  showing 275
EXIT 185
Exit command 532
Exit Undertow command 529
EXP 189

EXP10 189
EXP2 189
Expand All Drivers
  Schematic Window 246
Expand All Drivers command 336
Expand All icon 330
Expand All Loads
  Schematic Window 246
Expand All Loads command 336
Expand Buses button 523
expanding
  buses 80
Explode Bus button 80
Explode Bus icon 67, 295
exploding
  buses 80
exponent assignment operator 365
exponent operator 365
exposing
  panes 90
expressions 128
  analog 178
  arbitrary 230
  editing 220
  operators 316
extensions
  analog 390
Eye Diagram Analyzer 171
Eye Diagram Signals window 133, 171
eye diagrams
  signals 133
F
Fall Time Analyzer 173, 174, 220
Fast File 69
Fast file 283, 456, 462
Fast File format 35
Fast Files
  creating 69, 135
  generating 139
  loading 69
  Vdump 137
Fast files 119, 239, 280
Fast Fourier Transform (see FFT)
Fast Mode Flush Interval field 305
Fast Selection command 55
Fast Selection menu 53
FEATURE lines
  licenses 20
features
  analog 140
FFT 189, 221
fields
  Add Color To List 97, 148
  Alias 85
  Analog Expression 190
  Assign Scales 183
  Black & White Only 75
  Border 75
  Bundle Name 47
  Color 182
  Connectivity File 305
  Create Connectivity File 305
  Cursors Significant Digits 99, 150
  Dump Trace Info 305
  dumpFileName 293
  dumpTraceInfo 293
  End Time 75, 172
  Fast Mode Flush Interval 305
  File Suffix 126, 134, 337
  Foldover Time 171
  Font Size 75
  hdllintFileName 293
  Hide 90
  High Thresh 172, 174
  Library 521
  List 90
  Low Thresh 172, 174
  Monitor All Signals 305
  Number of Signals per Page 75
  Orientation 75
  Page Size 75
  Panel Title 90
  Print To 75
  Project Directory 305
  Project Name 305, 488
  Properties for Entire Overlay 183
  Run Directory 521
  Scale Options 75
  Show Names 75
  Show Pane Titles 98, 149
  Show Values 75
  Signal 85
  Signal Dump File 305
  Signal Line Size 75
  Signals to Use the Value Aliases 49
  simCmdLine 293

simCompileLine 293
Start Time 75, 171
Stop Time 171
Tag Significant Digits 99, 150
Title 75
Trace Info File 305
traceFileName 293
Undertow Executable 522
Undertow Host 522
Use Project Name For Files 305
Value 45
Value Aliases 49
Values Significant Digits 99, 150
verExecutable 293
Waveform Background Color 97, 148
Waveform Spacing 98, 149
Waveform Starting Offset 98, 149
Working Directory 126, 134, 337
Y-Axis Scale Significant Digits 99, 150
File dialog box 69
File Group button 64
File menu 67, 125, 134, 178, 299, 332, 346
file menu (source code viewer) 299
file open (source code viewer) 299
File Open menu 248, 300, 333
File Print menu 334, 347
file save (source code viewer) 299
file save as (source viewer) 299
File Save menu 302
File Selection dialog 64
File Suffix command 87, 178
File Suffix field 126, 134, 337
file_types 253
files
3DB.script 391
AC 523
Alias 87
Amplitude.script 391
analog_scripts.tar 391
AtX.script 391
Average.script 391
base 277
Baseline.script 391
cds.lib 534
cdsinit 516, 517
closing 70
comparing 31, 51
creating 76
cshrc 516
DC 523
def 348
DeltaX.script 391
DeltaY.script 391
design 464
dump 139, 484, 533
DutyCycle.script 391
EPS 229
Fast 119, 239, 280, 283, 456, 462
Fast File 35
FLI 280, 284
foreign.vhdl 464
iv.v 457, 507
license.dat 20
loading 248, 294
Makefile.nc 468
MaxFall.script 392
MaxPoint.script 392
MaxRise.script 392
Memory Dump 489
MinPoint.script 392
multiple 36
ncelab.args 469
ncsim.args 469
opening 251
output 31, 51, 229
overlay 277
PeakToPeak.script 392
PLI 280, 283
PostScript 230
probe 528
Project 291
project 282
qtest.v 139
README_PLI 139
ScriptMenu.script 392
si.env 534
signal 253, 254, 283
signals 35
sigs 454, 457, 463, 468
SlewRate.script 392
source code 252, 268
source.vc 249, 251, 287
source_code.vc 283
tab 457
Topline.script 392
Trace 455, 485
trace 253, 254, 284

trace dump 455, 485, 486
transient 523
undertow.ile 516
Undertow.Init 515
Undertow.init 517
UndertwoAnalogExtension.pm 392
VCD 69, 119, 254, 280, 283, 522, 533
Verilog 283
verilog.dump 138
verilogI.ini 517
veriuser.c 497
VHDL 283
vhdl.dump 465
vt_finsimuser.c 507
vt_ptab 507
vt_vcs.tab 457
vt_veriuser.c 453, 454, 497
vtDump 457
vtpli.o 453, 454, 497
vtplifinsim.o 507
vtplimodtech.so 504
vtplivcs.o 457
vtrun.ascii 486
vtrun.ps 486
wav 284
wlf 284
Xdefaults 218, 372
Files dialog 301
Files panel 300, 301
Filter Icon
Source Code Window 238
Filter icon 296
filter_order parameter 194
filter_type parameter 194
filtering
Alias files 87
filters
analog 192
ELLIPTIC 195
FIR 194
IIR 194
syntax 193
Find All button 130
Find All command 132
Find command 312
Find List button 130
Find List command 132
Find Logic dialog 129
Find Logic window 130
Find Selected button 130
Find Selected command 132
Find Signals icon 296
FindFirstAndLastPeaks function 392
finding
connections 263
information 93
logic 129
schematic elements 273, 342
signal assignments 269
signals 218, 269, 325
Source Code 352
source code 268, 273, 275, 326, 342
FindNDBPoints function 393
FindNPeaks function 393
FindPointsOfValue function 393
FindSignals function 368
FinSim 507
Finsim 255
FinSim simulator 284
Fintronic FinSim 507
FINUSERPLIOBJ environment variable 507
finvc command 507
FIR 189
FIR filters 194
fit to pane
waveforms 113
Fit To Pane button 143
Fit to Pane button 219
Fit To Pane command 113
Fit To Pane icon 62, 160
Fix Cursor command 55, 108
flashing
widgets 200, 375
FLEXlm 17
license server 19
FLI
testbench 464
FLI files 280, 284
Floating Choose Window command 58, 124
Floating Group Window command 124
Floating Groups window 63
Floating Icon Bar command 124
Floating Menu Bar command 124
floating windows 124
Floating Windows command 52, 63

Floating windows command 53, 59
Fold over Time 133
Foldover Time field 171
Font Size field 75
fonts 217
  properties 92
Fonts command 54, 92, 217
Fonts dialog 93
for statement 364
foreign.vhdl file 464
Format menu 76
formats
  Fast files 239
  gunzip 19
  gzip 19
forms
  Options 520
  OSS Simulation Run Directory 520
forward
  stepping 267
Forward command 336
Forward Icon
  Source Code Window 236
Forward icon 296
forward icon (source code viewer) 296
Frame widget 198, 374
FREE 185
functionality
  plotting 214
Functions 366
functions 368
  ACOS 188
  ACOSH 188
  ACOSPI 188
  analog 134, 176
  ASIN 188
  ASINH 188
  ASINPI 188
  ATAN 188
  ATANH 188
  ATANPI 188
  AVERAGE 188
  Average 392
  BESSEL 188
  BUTTERWORTH 188
  CBRT 188
  CHEBYSHEV1 188
  CHEBYSHEV2 188
  CloseAllFilesAndOpen 367
  COEFPS 188
  CompareAndInsertInHash 397
  CompareLogicSignals 367
  COMPLEX 189
  COS 189
  COSH 189
  CreateAnalogSignal 367
  CreateNewVariable 367
  CreateXAxis 367
  CutSignals 367
  DB10 189
  DB20 189
  DeselectAllSignals 367
  DIFF 189
  DisplayVariableAddToOverlay 368
  ELLIPTIC 189
  EndTime 368
  ERF 189
  ERFC 189
  EXP 189
  EXP10 189
  EXP2 189
  FFT 189
  FindFirstAndLastPeaks 392
  FindNDBPoints 393
  FindNPeaks 393
  FindPointsOfValue 393
  FindSignals 368
  FIR 189
  GetAbsoluteValue 394
  GetCurrentXAxisIndex 368
  GetCursor 368
  GetDutyCycle 394
  GetFirstMaxPoint 394
  GetFirstMinPoint 394
  GetFirstSameValuePoints 394
  GetFrequency 395
  GetMaxFall 395
  GetMaxRise 395
  GetNextPointAfterCutoff 395
  GetPeriod 395
  GetPointOfValue 395
  GetPulseWidth 396
  GetSanitizedSignalValues 396
  GetSelectedSignals 368
  GetSelectedYAxisIndex 368
  GetSlewRate 396
  GetValueAtTime 368

GetXAxisCount 368
GetXAxisTitle 368
GetXAxisZoom 368
GetYAxisCount 368
GetYAxisSignalCount 369
GetYAxisZoom 369
GoToAtTime 396
GoToValue 369
IFFT 189
IIR 189
IMAG 189
InsertInIndexedHash 398
INTEG 190
LOG 190
LOG2 190
MAGN 190
MainLoop 369
MAX 190
MaxTimeX 369
MIN 190
MinTimeX 369
ModuleGetModules 369
ModuleGetVariables 369
NDBValue 398
OpenDumpFile 369
OpenProjectFile 369
PHASE 190
PlaceAnalogCursor 369
PlaceCursor 369
PlaceTagUsingCursors 369
PlaceTitleTag 370
PlaceValueTagUsingCursor 370
PLOTXY 190
POW 190
PrintSelectedSignalsToFile 370
PrintVarArray 397
REAL 190
RemoveXAxis 370
Restore 370
RMS 190
SelectSignal 370
SelectXAxis 370
SetXAxisTitle 370
SetYAxisZoom 370
SIN 190
SINPI 190
SortOutput 371
SQRT 190
TAN 190
TANH 190
TANPI 190
TestAanalyzerSetTimingSignal 371
VariableGetIncrementalValue 371
VariableGetIncrementalValueAfterCutoff 397
VariableGetType 371
VariableGetValues 371
VariableSetValues 371
Zoom 371
Functions Calculator 186
Functions list 179
G
g 166
Gate browser 295, 327
gate-level 261
gate-level schematics 244
generating
dump files 138
Fast Files 139
Get Information command 93
Get Information Dump Signals command 94, 95
GetAbsoluteValue function 394
GET-COLOR command 539
GetCurrentXAxisIndex function 368
GetCursor function 368
GetDutyCycle function 394
GetFirstMaxPoint function 394
GetFirstMinPoint function 394
GetFirstSameValuePoints function 394
GetFrequency function 395
GetMaxFall function 395
GetMaxRise function 395
GetNextPointAfterCutoff function 395
GetPeriod function 395
GetPointOfValue function 395
GetPulseWidth function 396
GetSanitizedSignalValues function 396
GetSelectedSignals function 368
GetSelectedYAxisIndex function 368
GetSlewRate function 396
getting
information 93
GetValueAtTime function 368
GetXAxisCount function 368
GetXAxisTitle function 368
GetXAxisType 368

GetXAxisType function 368
GetXAxisZoom function 368
GetYAxisCount function 368
GetYAxisSignalCount function 369
GetYAxisZoom function 369
Go Back button 45, 61
Go To button 45, 61
Go To command 229
Go To Cursor command 108
Go To Errors 127
Go to State command 348
Go To Value 129
GoTo
  dialog box 44
GoTo command 44
GoTo Error command 61
GoTo Icon
  Waveform Window 240
GoTo icon 61, 312, 355
GoTo LE command 61
GoTo Marker command 61
GoTo menu 248, 251
GoTo TE command 61
GoTo Time command 61
GoTo tool 50
  aliases 50
GoTo Transition command 61
GoTo Value command 61
GoToAtTime function 396
Go-to-State Icon
  State Diagram Window 243
Go-to-State icon 345
GoToValue function 369
Graphical User Interface (see GUI)
graphs
  output 229
greater than operator 365, 366
greater than or equal to operator 366
grid
  horizontal 100, 151
  vertical 101, 152
Grid Signal command 108
Grid Signal LE command 108
Grid Signal TE command 108
grids 217
  horizontal 100, 151
  signals 108
  vertical 101, 152
Grids and Scale command 183
Grids and Scales command 217
Group Definition button 64
Group Icon
  Waveform Window 240
grouping
  signals 46, 63, 81, 218
Groups button 218
Groups icon 59
GUI 217
gunzip
  format 19
gzip format 19
H
HDL
  text window 527
HDL Analyzer 264
HDL menu 526
HDLLint 312
HDLLINT Report command 323
hdllintFileName field 293
height
  waveforms 41, 122
Heirarchical Browser 237
HELP 185
Help button 64
Help command 178
Help Icon
  Source Code Window 237
  Waveform Window 241
hexadecimal
  annotating 318
Hexadecimal command 76
Hide field 90
hiding
  panes 90
hierarchy
  descending 529
  modules 37
  multiple levels 264, 338
Hierarchy Browser 531, 532
Hierarchy icon 295
Hierarchy Location tab 296
Hierarchy menu 529
High Thresh field 172, 174
Highlight command 207, 383

Highlight Errors 127
Highlight Errors command 126
Highlight Found button 130
Highlight Net
  Schematic Window 246
Highlight Net button 263
Highlight Net command 336
Highlight Net icon 245, 273, 330
highlighting
  nets 273, 342
histograms 167
History command 310
horizontal
  scrolling 115
Horizontal and Vertical Offset command 183
horizontal grid
  pixels 100, 151
horizontal panes 267, 349
horizontal scroll bar 118
HSpice 15, 31, 51
I
icon bar 53
icons
  Actions 67
  All 67, 295
  analog 141
  Analog Configuration 62, 183
  Auto Update Off 118
  Back 296
  Breakpoint On Line 355
  Breakpoint On Signal 355
  Choose 58
  Chooser 66
  chooser 146
  Chooser toolbar 66
  Coincidence 352
  Copy 54, 56, 67, 295
  Cut 54, 56
  Display 58, 67, 295
  Display Current Drivers 296, 355
  Down 297, 330
  Error 127
  Expand All 330
  Explode Bus 67, 295
  Filter 296
  Find Signals 296
  Fit To Pane 62, 160
  floating 124
  Forward 296
  GoTo 61, 312, 355
  Go-to-State 345
  Groups 59
  Hierarchy 295
  Highlight Net 245, 273, 330
  Inputs 295
  Inspected Selected Signal 355
  Measure Tag 62
  Measures 161
  Modifies Selected Signal 355
  Modules 296
  Open A Source File 355
  Option 277
  Options 67
  Outputs 295
  Overlay Files 295
  Paste 54, 57
  Paste Blank 57
  Persistence 275, 345, 352
  Scroll To 57
  Search Backward 297, 331
  Search Forward 297, 331
  Search Signal 297
  Search String 297
  Select All 56, 91
  Set Scope 354
  Show Details 312
  Show Drivers 296
  Show Entire Schematic 271, 281, 330
  Show Filenames 295
  Show Loads 297
  Show Net 281
  Show Status 355
  Snap 62
  Snap Source 62
  source code 295
  Split Sweeps 295
  Start Interactive 534
  State Coincidence 275
  State Coincidence, Display 345
  State Coincidence, Step 345
  State Coverage 345
  State Sequence 345
  State Table 275, 345
  state window 274
  Step 345
  Step Backward 274, 297, 345, 350, 354
  Step Forward 274, 297, 345, 350, 354

Stop 271
Stop Expansion 330
Tag 62
Text 67, 295
toolbar 56
Trace Window 354
Turn Animation On 354
Undo 57
Up 297, 330
Update 60, 118
Value Tag 62
windows 124
Zoom Full 59, 330, 344
Zoom In 59, 330, 344
Zoom One to One 330, 344
Zoom Out 59, 330, 344
if statement 364
IFFT 189
Ignore Initial ‘x’ button 130
IIR 189
IIR filters 194
IMAG 189
importing design 300
importing source code 300
inequality 364
inequality operator 366
information
getting 93
signals 93
Inoise 221
input_coeff_list parameter 194
input_signal parameter 194, 195
Inputs icon 295
InsertInIndexedHash function 398
Inspect Signal dialog 321
Inspect Signal window 314
Inspected Selected Signal icon 355
installing 15
disk space 18
Undertow 18
INTEG 190
integration
Verilog 530
integration kit 515
interactive mode 280
interactive simulation 250
opening files 251
Interactive Simulations 290
Interactive tool 70, 481
interactive tool 36
Interactive Tool menu 302
Interactive_tool 328
Interpolate Current command 168
Interpolate Voltage command 168
interpolating
current 132
voltage 132
interpreted simulators 250
Interrupt command 310
intervals
3X Current Time 118
5X Current Time 118
All Time 118
Current Display Time 118
display 117
End 118
Minimum Time 118
Start 118
Iv parser 488
iv.v file 457, 507
ivsimcmp command 255
K
Kill command 310
L
label statements 363
Label widget 198, 202, 374, 377
labeling
list-boxes 207, 383
labels
adding 206, 213, 382, 388
creating 202, 377
last statement 363
leading edge
searching 297, 331
trailing 297
LEAPFROG 15, 31, 51
left scale 101, 152
less than operator 364, 366
less than or equal to operator 366
levels
tracing 487
libraries
Cell 521
View 521
Library field 521

licence.dat file 20
license server 19
license.dat file 20
licenses 17
  DAEMON lines 20
  environment variables 21
  FEATURE lines 20
  lmgrd command 21
  lmstat command 21
  software 17
licensing
  permissions 20
linear axis 215
Link Cursors command 109
linking
  cursors 109
  PLI 491
  Undertow 535
LIST 185
List (expressions) command 128
List Box widget 198, 374
List button 126
List command 81, 85
List field 90
List of Signals to Ignore button 130
List of Signals to Ignore command 132
list-boxes
  adding 207, 382
  labeling 207, 383
  Templates 206, 207, 382
  Widgets 206, 207, 382, 383
listing
  open windows 107
lists
  Functions 179
  Operators 134, 178, 187, 190
  signal name 295
LM_LICENSE_FILE 19, 21, 247
  environment variable 20
LMACRO 185
lmgrd 19
lmgrd command 21
lmstat 19
lmstat command 21
Load Additional File command 275, 489
Load Batch Run command 69, 312, 488
Load Step File command 348
loading
  breakpoints 314
  data 117
  Fast Files 69
  modules 349
  source code 325
loading files 294
  menus 248
loading wave forms (source code viewer) 300
loads
  displaying 257, 258
locating
  schematic elements 273, 342
  signals 218, 325
  Source Code 352
  source code 268, 273, 275, 326, 342
Lock Cursors command 109
Lock X At First Point button 104, 155
Lock X At Value button 106, 157
Lock Y At First Point button 105, 156
Lock Y At Value button 107, 158
locking
  cursors 108, 109
LOFF 185
LOG 190
log axis 215
LOG2 190
logic
  finding 129
Logic Analyzer 128
Logic command 77
Low Thresh field 172, 174
LPR_CALL variable 75
M
MACRO 185
MAGN 190
magnitude
  results 221
Main Menu Bar 53
main toolbar 56
main window
  Undertow 32, 52
MainLoop function 369
Makefile.nc file 468
Manual printing 74
markers
  time 53

MAX 190
MaxFall.script file 392
MaxPoint.script file 392
MaxRise.script file 392
MaxTimeX function 369
measure tag 43
Measure Tag button 145
Measure Tag icon 62
measure tags 43
  waveforms 113
Measure Tags command 113
measurements
  boundary 210, 385
Measures button 124, 142
Measures icon 161
measuring
  analog signals 208, 384
memory
  viewing 489
  viewing contents 275
Memory Dump file 489
MENTOR 15, 31, 51
menu
  Fast Selection 53
menu bar 52
Menubar 206, 382
  Builder window 211, 386
Menubar command 210, 386
menus
  Actions 295, 311
  Analog Functions 220
  Clear Errors 126
  Details 318
  Edit 125, 134, 178, 335, 347
  Fast Selection 53
  File 67, 125, 134, 178, 299, 332, 346
  File Open 248, 300, 333
  File Print 334, 347
  File Save 302
  floating 124
  Format 76
  GoTo 248, 251
  HDL 526
  Hierarchy 529
  Interactive Tool 302
  loading files 248
  Opening Files 248
  Option 295
  Options 78, 126, 335, 348, 532
  Save As 302
  Schematic 331
  Search Signal 331
  Sessions 303
  Simulation 522
  Simulator 310, 318
  Source Code 298
  starting Undertow 34
  State 345
  Tools 125, 166, 323
  Undertow 517
  Wave Window 98, 149
  Window 96, 134, 178, 313, 337, 349
  Windows 45, 126
  windows 124
menus (source code viewer) 298
menus window 58
middle mouse button 233
MIN 190
minimum pixels
  horizontal 100, 151
  vertical 101, 152
Minimum Time interval 118
MinPoint.script file 392
MinTimeX function 369
ModelTech 15, 31, 51
Modeltech
  PLI 462
ModelTech environment variable 464
ModelTech Verilog 255
ModelTech VHDL 255
Modeltech Vsim 504
modes
  compression 483
  dynamic 498
  Real Time 115
  static 498
Modifies Selected Signal icon 355
Modify button 64
Modify Group window 64
Modify Selected (expression) command 128
Modify Selected command 81, 85
Modify Selected Signal command 313
modifying
  breakpoints 314
Module 269
Module browser 328

module browser 262
Module button 280
Module Icon
  Source Code Window 237
module list window 58
Module pane 269, 270
ModuleGetModules function 369
ModuleGetVariables function 369
modules
  browser 36
  hierarchy 37
  loading 349
  UndertowAnalogExtension.pm 397
  viewing 262
Modules button 269, 325
Modules icon 296
monitor all signals (source code viewer) 305
Monitor All Signals field 305
Monitors and Dumps button 488
Monitors and Dumps command 488
mouse
  zoom 40
mouse controls 232
  box zoom 233
  click 233
  double click 233
  drag and drop 233
  middle mouse button 233
  right click 233
  select 233
moving
  cursors 108, 124
MTI
  compiling 465
  simulating 465
multibranches
  VCT 260
multiple axis
  plotting 215
multiple files 36
multiple levels of hierarchy 264, 338
multiple panes 45, 267
multiple scopes 264, 338
multiple views 349
N
Name Alias command 81, 85
Names command 336
naming
  signals 80
Navigator 107
Navigator command 107, 313
NC Verilog 499
  PLI 468
ncelab program 469
ncelab.args file 469
ncprep program 469
ncsim program 469
ncsim.args file 469
NC-Verilog 255, 515
ncvlog program 469
NDBValue function 398
negate operator 365
net colors 522
netlists
  Antrim 534
  bit 518
  buses 518
  connectivity 51
  tracing 31
  Verilog 534
nets
  highlighting 273, 342
New (expression) command 128
new alias dialog 49
New Bundle dialog 47
New Command window 199, 374
New Group button 64
NEW signal 185
New Source Code command 323
New Text window 200, 375
New Waveform command 323
new_value_index 398
next statement 363
No Tag Lock button 104, 155
non-compiled simulators 250, 253, 284
Normal command 76
Normalize zoom 161
not equal operator 364
NOT operator 364
null list 365
Number of Signals per Page field 75
numeric equal operator 364
O
obtaining software

Undertow 17
web 17
Octal command 76
offsets
axis 215
One button 126
One Shot 316
One Undertow Only button 524
Ones Compliment command 76
Onoise 221
OPEN 185
Open A Source File icon 355
Open Additional File command 36, 68
Open command 69, 70, 299, 332, 346, 517
Open Dump File dialog 135
Open File 35
open file (source code viewer) 299
Open Source dialog 301
Open Waveform command 69
Open Window command 314, 318
open windows 107
Open/Close command 135
Open/Close File command 68, 219, 249, 252, 287
OpenAdditonalFile command 276
OpenDumpFile function 369
OpenFiles command 276
opening files
interactive simulation 251
Opening Files menu 248
OpenProjectFile function 369
operators
AND 364
auto-decrement 365
auto-increment 365
comparison 365, 366
concatenation 365
concatenation assignment 365
exponent 365
exponent assignment 365
expressions 316
greater than 365, 366
greater than or equal to 366
inequality 366
less than 364, 366
less than or equal to 366
negate 365
NOT 364
null list 365
numeric equal 364
OR 364
Perl 364
range 366
repetition 366
repetition assignment 366
string equality 366
strings 366
Operators list 134, 178, 187, 190
operatorsnot equal 364
Optimizing tool 481, 483
Optimizing_tool 119, 139
options
command line 247
simulator 250
source code 250
Options form 520
Options Icon
Source Code Window 237
Options icon 67, 277
Options menu 78, 126, 295, 335, 348, 532
OR operator 364
Orientation field 75
OSS Simulation Run directory 518, 530
OSS Simulation Run Directory form 520
output
graphs 229
recalling 95
simulation 118
Output dialog 94
Output Display window 275
output files 229
viewing 31, 51
Output window 94, 162
output_coeff_list parameter 194, 195
Outputs icon 295
Over Draw Mode command 348
Overlay Analog Signal 164
Overlay Analog Signals 169
Overlay Analog Signals window 132, 163, 168
Overlay dialog 92
overlay file 277
Overlay Files command 92
Overlay Files icon 295
Overlay Waveforms command 163, 168

overlaying
signals 147, 276
waveforms 132
overlays
properties 91
P
Page Size field 75
Pane Controller window 90
Pane Time Zoom 161
Pane Zoom Analog command 160, 168
Pane Zoom window 91
Panel Title field 90
panels
Directories 300
Files 300, 301
panes 45
adding 46, 122, 267
adding value areas 123
Arbitrary X 46, 180, 192, 195
controlling 90
Current 90, 160
exposing 90
fitting waveforms 113
hiding 90
horizontal 267, 349
Module 269, 270
multiple 267
Power 90, 160
removing 123
removing value areas 123
Signal Selector 270
signal selector 269
synchronizing 122
Time 46, 90, 160, 180, 192, 195
vertical 267, 349
Voltage 90, 160
zooming 90
panning
waveforms 41
parameters
filter_order 194
filter_type 194
input_coeff_list 194
input_signal 194, 195
output_coeff_list 194, 195
pass_limit2 194
pass_ripple 194
sample_freq 194, 195
stop_limit 194
VtDumpvars 482
parametric sweep 221
parsers
Iv 488
pass_limit1 parameter 194
pass_limit2 parameter 194
pass_ripple parameter 194
Paste Blank Icon
Waveform Window 240
Paste Blank icon 57
Paste button 87
Paste command 335, 347
Paste Icon
Waveform Window 240
Paste icon 54, 57
Paste, Add To Overlay button 141
Paste, New Overlay button 141
PATH 22
PATH environment variable 490
path environment variable 456, 468
patterns 216
PAUSE 185
PeakToPeak.script file 392
Perl
comments 362
operators 364
Perl code
adding 203, 378
Perl functions (see functions)
Perl language 362
Perl script
example 371
Perl scripts
creating 201, 377
Perl scripts (see scripts)
Perl statements (see statements)
PERLLIB 22
permissions
licensing 20
Persistence command 348
Persistence Icon
State Diagram Window 243
Persistence icon 275, 345, 352
Persistence tool 281
PHASE 190
pixels

divisions 101, 152
horizontal 100, 151
horizontal grid 100, 151
vertical 101, 152
vertical grid 101, 152
waveform height 41
PlaceAnalogCursor function 369
PlaceCursor function 369
PlaceTagUsingCursors function 369
PlaceTitleTag function 370
PlaceValueTagUsingCursor function 370
placing
cursors 124
PLI
compiling 463
dynamic mode 498
examples 484
linking 491
Modeltech 462
NC Verilog 468
simulating 463
testbench 456, 462, 468
VCS 456
PLI directory 457
PLI files 280, 283
PLI static command 453
PLI tools 481
PLI_BUILD environment variable 468
pli_example_files directory 139
PLIOBJS environment variable 463, 464, 465, 504
Plot Bundle Net button 523
Plot Signal Once button 524
plots
sizing 217
synchronizing 218
plotting
axis 215
functionality 214
waveforms 214
PLOTXY 190
Pop Up Signals command 527
PostScript files 230
POW 190
Power pane 90, 160
POWERMILL 15, 31, 51
PRINT 185
Print command 332, 346
print command 362
Print dialog 74, 75
Print Signals command 230
Print To field 75
Print To File command 334, 347
Print To Printer command 334, 347
printing
Automatic 74
Manual 74
signals 73
WYSWYG 74
PrintSelectedSignalsToFile function 370
PrintVarArray function 397
probe files
saving 528
probing
signals 532
programs
cr_inca 469
dumpvars 119, 138, 139, 456, 468, 482
ncelab 469
ncprep 469
ncsim 469
ncvlog 469
simv 457
vccdump 507
vconfig 453, 454, 497, 498
vdump 69, 119, 135, 137, 456, 468, 490
vsim 465
VTCOMPRESS 468
vtDump 119, 139, 287, 455, 456, 457, 462, 463, 468, 490
vtdump 119
vtDumpvars 139, 287, 454, 456, 457, 463, 468, 482
vtrun 455, 485
xrdb 217
Project Directory field 305
Project file 291
project file 282
Project Name field 305, 488
projects - opening a project 300
Projects command 300, 302
properties
analog waveforms 181
fonts 92
overlays 91

Properties command 91
Properties for Entire Overlay field 183
Properties for Signal dialog 182
Pulse Width Analyzer 175
Pulse Width Analyzer dialog 176
Q
qtest.v file 139
Quick Command Bar 134, 178
Quick Command bar 87
QUIT 185
quit (source code viewer) 299
Quit command 299
R
Radio Button widget 199, 374
range operator 366
reading
  dumpfiles 70
README_PLI file 139
REAL 190
Real Time command 118
Real Time mode 115
recalling
  output 95
redo statement 363
Register Transfer Level (see RTL)
regular expressions 218, 296
Remove All Probes command 526
Remove HDL Selection command 528
Remove Instance Module Nets command 526
Remove Net command 525
Remove Pane command 123, 349
Remove Tag command 112
RemoveXAxis function 370
removing
  breakpoints 324
  panes 123
  value areas 123
repetition assignment operator 366
repetition operator 366
Reread Dumpfiles command 215
Reset command 310
Restore command 65, 71, 230
Restore function 370
restoring
  saved files 71
results
  arbitrary expressions 230
  magnitude 221
  viewing 456, 457, 465, 469
Results dialog 318
Results window 202, 205, 207, 309, 314, 377, 380, 383
retrieving
  output 95
right click 233
right scale 101, 152
Rise Time Analyzer 172, 173, 220
RMS 190
root
  tracing 487
RTL 244, 261, 281
Run command 310, 464, 488
Run directory 518
run directory 196, 361
Run Directory field 521
RUN filename 185
Run HDLLINT command 312
S
sample_freq parameter 194, 195
save - source code viewer 300, 302
save (source code viewer) 299
Save All command 71, 72
save as (source code viewer) 299
Save As command 204, 205, 214, 299, 332, 338, 346, 379, 380, 389
Save As menu 302
Save command 65, 71, 218, 299, 302, 332, 346
Save File command 333
Save Groups button 65
Save Groups command 65, 71
Save Probe File command 528
saved files
  restoring 71
saving
  breakpoints 314
  scripts 204, 214, 379, 389
  sessions 71
  templates 205, 380
scalars
  variables 362
Scale Options field 75
scales
  left 101, 152

right 101, 152
time 55
scaling 74
Schematic Browser
starting 328
Schematic Browser command 128
Schematic command 261, 271, 272, 323, 328, 341
schematic elements
finding 342
locating 273
Schematic menus 331
Schematic Tool button 62
Schematic toolbar 329
Schematic Viewer 262
Schematic Window
Down Icon 246
Expand All Drivers 246
Expand All Loads 246
Highlight Net 246
Search Backward Icon 246
Search Forward Icon 246
Search Selector 246
Search String 246
Show Drivers Icon 245
Show Loads Icon 245
Show Whole Schematic 246
Stop Layout 246
Zoom 1 to 1 Icon 243, 245
Zoom Full Icon 245
Zoom In Icon 245
Zoom Out Icon 245
Schematic window 128, 261, 263, 266, 272, 273, 281, 327, 336, 342, 343, 531, 532
schematic window 244
highlighting nets 273
starting 271
tracing signals 272
Schematic Window tool 62
schematics
displaying signals 273
finding source code 273
gate 244
RTL 244
viewing 261, 266
scopes
multiple 264, 338
Script command 200, 376
Script Tool 196, 359, 372
Script tool command 205, 381
Script Tool window 183, 196, 202, 210, 361, 377, 386
Script Type command 204, 205, 214, 379, 380, 389
ScriptMenu.script file 392
Scripts 198, 374
scripts
attaching to buttons 213, 389
cr_inca 468
creating 201, 377
saving 204, 214, 379, 389
Scripts command 201, 210, 377, 385
scripts directory 392
Scroll 90% command 117
Scroll Bar widget 199, 374
scroll bars 55
horizontal 118
Scroll LE command 115
Scroll Percent command 116
Scroll Set Interval command 117
Scroll TE command 116
Scroll To Icon
Waveform Window 240
Scroll To icon 57
Scroll To Max Time command 116
Scroll To Signal command 57, 117
scrolling
horizontal 115
vertical 117
Search Backward Icon
Schematic Window 246
Source Code Window 236
Search Backward icon 297, 331
Search dialog 319, 320
Search Forward Icon
Schematic Window 246
Source Code Window 236
Search Forward icon 297, 331
Search Selector
Schematic Window 246
Source Code Window 236
search signal (source code viewer) 297
Search Signal icon 297
Search Signal menu 331
Search String

Schematic Window 246
Source Code Window 236
search string (source code viewer) 297
Search String icon 297
Search window 314
searching
connections 263
information 93
leading edge 297, 331
logic 129
schematic elements 273, 342
signal assignments 269
signals 218, 269, 325
Source Code 352
source code 273, 275, 326, 342
strings 331
trailing edge 331
transitions 297, 331
values 297, 331
select 233
Select All button 79, 87
Select All command 335, 347
Select All Icon
Waveform Window 240
Select All icon 56, 91
Select Overlay Properties 182
Select Overlay Properties window 181
Selected Analog Waveform command 180, 192, 195
Selected Analog Waveform Properties command 161, 181, 215, 216
Selected Analog Waveform window 182
Selected button 126
Selected Signal 172, 173, 175
Selected Signals window 79, 80, 87
selecting 274
axis labels 217
colors 216
fonts 217
grids 217
patterns 216
signal files 35
signals 218, 539
sizes 217
source code 268
styles 217
tics 217
titles 217
value labels 217
selecting (see setting)
SelectSignal function 370
SelectXAxis function 370
separating
windows 124
Sequence Analyzer 132
sequences
time points 132
sessions
saving 71
starting 72
synchronizing 72
Sessions menu 303
sessions menu - source code viewer 303
Set Breakpoint On Selected Signal command 313
Set Default command 122
Set Display Interval command 117
Set Master Signal button 142
Set Scope button 79
Set Scope icon 354
Set Selected button 122
Set Tag Text command 112
setting
Active View 350
breakpoints 316, 324
cursors 108
font properties 92
overlay properties 91
time units 120
variables 203, 378
waveform height 122
settrace command 486
setup 15
demo licenses 19
SetXAxisTitle function 370
SetYAxisZoom function 370
Show Connectivity command 128
Show Connectivity window 128
Show Details icon 312
Show Driver command 309
Show Drivers 318
show drivers (source code viewer) 296
Show Drivers button 257
Show Drivers command 336
Show Drivers Icon

Schematic Window 245
Source Code Window 236
Show Drivers icon 296
Show Entire Schematic icon 271, 281, 330
Show Filenames icon 295
Show Hierarchal Names command 128
Show Histogram command 167
Show Loads 318
show loads (source code viewer) 297
Show Loads command 309, 336
Show Loads Icon
Schematic Window 245
Source Code Window 236
Show Loads icon 297
Show Names field 75
Show Net icon 281
Show Pane Titles field 98, 149
Show Selector command 206, 337, 349, 381
Show Selector window 338
Show Status icon 355
Show Values field 75
Show Whole Schematic
Schematic Window 246
Show Whole Schematic command 336
showing
coincident states 275
exercised states 275
si.env file 534
sigfile 253, 254, 283, 284
Signal 218
signal assignments
finding 269
Signal Browser window 67
Signal Dump File field 305
Signal field 85
signal files 253, 254, 283
selecting 35
Signal Line Size field 75
Signal List window 79
signal list window 58
Signal Name Area 241
signal name list 295
Signal Name window 45
Signal Selector 238, 280
signal selector 281, 294
Signal Selector pane 270
signal selector pane 269
Signal Selector window 245, 260, 273
Signal To Expression 317
Signal Values Area 241
SIGNALS 185
Signals 147
signals
adding 537
aliases 48, 50
analog 96, 134, 140, 147, 179, 208, 384
Auto Update 118
Boolean functions 128
bundled 48
bundling 46, 81
clock 238
colors 216, 539
comparing 127
Ctrl 46
data 220
deleting 532, 538
display all 39
displaying 273, 326, 342, 517
displaying from source code 270
edges 44
eye diagrams 133
finding 269, 325
grids 108
grouping 46, 63, 218
information 93
names 53
naming 80
overlaying 147, 276
patterns 216
printing 73
probing 532
scaling 74
searching 218
selecting 218, 539
tracing 272, 341
updating 118
values 44, 54
viewing 38, 531
Signals Found window 296
Signals to Use the Value Aliases field 49
sigs file 454, 457, 463, 468
sigt command 211, 387
sigthresh command 212, 387
simCmdLine field 293
simCompileLine field 293

Simulate for a Time command 355
simulating 455, 457
  MTI 465
  PLI 463
simulation
  Analog AC 522
  Analog DC 522
  Analog Transient 522
  batch 252, 283
  interactive 250, 251
  output 118
  Real Time 118
  results 456, 457, 465, 469
  Verilog 522
  Verilog-AMS Transient 522
  virtual 260
Simulation menu 522
simulator
  options 250
  starting 488
  stepping 280
Simulator Executable attribute 304
Simulator Executable window 250, 251
Simulator menu 310, 318
Simulator Options command 250
simulator options menu 303
simulator output files 304
Simulator Output Options window 304
Simulator Type attribute 304
Simulator Type command 250
simulator types 253, 255
simulators 31
  Antrim 522
  compiled 250, 254, 285
  FinSim 284
  interpreted 250
  non-compiled 250, 253, 284
  Spice 221
  supported 15, 51
  types 250
  VCS 284
  Verilog XL 284
  Verilog-XL 253
simv program 457
SIN 190
Single Scope Only command 336
SINPI 190
size
  plots 217
  windows 160
sizes 217
SlewRate.script file 392
slopes
  waveforms 133
Snap Icon
  Waveform Window 240
Snap icon 62
Snap Source Icon
  Waveform Window 240
Snap Source icon 62
Snap to Source Code button 271
Snap/Source command 261
snapping
  source code 261
  waveforms to source code 261
software
  installing 15
  licenses 17
  obtaining 17
  obtaining on web 17
  setup 15
SortOutput function 371
Source 294
Source Code
  finding 352
  window 294
source code
  breakpoints 258
  displaying 325
  displaying from signals 270
  displaying from waveforms 270
  finding 273, 275, 326, 342
  loading 325
  locating 268
  options 250
  schematic elements 273
  selecting 268
  snapping 261
  stepping 256
  Time column 323
  toolbar 257
source code - actions icon 295
source code - all icon 295
source code - back icon 296
source code - display current drivers icon 296
source code - display icon 295

source code - forward icon 296
source code - importing 300
source code - modules icon 295
source code - options icon 295
Source code - search signal 297
source code - show drivers icon 296
source code - show loads 297
source code - step backwards 297
source code - step forward 297
source code - window icons 295
Source Code Browser 236, 280
Source Code browser 260, 286, 322
source code browser 294, 325
Source Code command 248, 251, 268, 286, 300
source code files 252
Source Code Files/Simulation Files command 463
Source Code Files/Simulator Options window 303
source code icons 295
Source Code menus 298
source code menus 298
Source Code Module browser 271
Source Code Viewer 262
source code viewer 318, 351
Source Code Window 235
  Actions Icon 237
  All Icon 237
  Back Icon 236
  Display Current Drivers Icon 236
  Display Icon 237
  Down Icon 236, 246
  Filter Icon 238
  Forward Icon 236
  Heirarchical Browser 237
  Help Icon 237
  Module Icon 237
  Options Icon 237
  Search Backward Icon 236
  Search Forward Icon 236
  Search Selector 236
  Search String 236
  Show Drivers Icon 236
  Show Loads Icon 236
  Steps Backward Icon 236
  Steps Forward Icon 236
  Up Icon 236
  Variables Icon 238
Source Code window 66, 235, 243, 245, 256, 261, 270, 272, 273, 280, 281, 286, 301, 302, 309, 312, 313, 319, 343, 344, 352, 405, 440
  Signal Selector 238
source code window 235, 295
source code/simulator options menu 303
Source Window 233
Source window 258, 324, 328, 464, 489
  breakpoints 323
Source Window command 318
source.vc file 249, 251, 287
source_code.vc file 283
spacing
  waveforms 91
Spice simulator 221
Split Sweeps icon 295
SQRT 190
Start Interactive icon 534
Start Interactive Tool command 36
Start interval 118
Start New Undertow command 72, 218
Start Time 133, 172, 174, 175
Start Time command 131
Start Time field 75, 171
Start Undertow command 525, 531
starting
  Builder 206, 381
  new sessions 72
  Schematic Browser 328
  schematic window 271
  simulator 488
  State Tool 344
  state window 273
  Undertow 520, 535
  Undertow Suite 246, 282, 286
  Undertow suite 252
starting Undertow 32
  command line 33
  menus 34
starting Undertow Suite 291
state
  bubble 275
State and Schematic command 274
State Browser
  stepping 351

State browser 344, 351
State Coincidence Display Icon
  State Diagram Window 243
State Coincidence icon 275
State Coincidence Step Icon
  State Diagram Window 243
State Coincidence tool 281
State Coincidence, Display command 348
State Coincidence, Display icon 345
State Coincidence, Step command 348
State Coincidence, Step icon 345
State command 274
State Coverage command 348
State Coverage Icon
  State Diagram Window 243
State Coverage icon 345
State Diagram
  stepping 350
State Diagram command 323
State Diagram tool 62
State Diagram Window
  Go-to-State 243
  Persistence Icon 243
  State Coincidence Display Icon 243
  State Coincidence Step Icon 243
  State Coverage Icon 243
  State Sequence Icon 243
  State Table Icon 243
  Steps Backward 243
  Steps Forward 243
  Zoom In Icon 243
  Zoom Out Icon 243
State Diagram window 266, 274, 280, 344, 350
state diagram window 242
state diagrams 274
  selecting 274
  stepping 274
  viewing 266
State Machine browser 343, 348
State menus 345
State Sequence command 348
State Sequence Icon
  State Diagram Window 243
State Sequence icon 345
state table
  displaying 275
State Table command 348
State Table Icon
  State Diagram Window 243
State Table icon 275, 345
State Table tool 281
State Tool
  starting 344
State Tool button 62
State toolbar 344
State Viewing window 346
State window 261, 266
  adding panes 267
state window
  finding source code 275
  icons 274
  starting 273
statements
  continue 364
  for 364
  if 364
  label 363
  last 363
  next 363
  redo 363
  until 364
  while 363
states
  coincident 275
  exercised 275
States and Schematic command 261, 266, 271
States and Schematics command 328
States command 266
static mode
  Verilog-XL 498
status
  commands 259
  VCT branch 259
  VCT multibranch 260
Step Back command 348
step backward (source code viewer) 297
Step Backward icon 274, 297, 345, 350, 354
step files
  loading 348
step forward (source code viewer) 297
Step Forward command 348
Step Forward icon 274, 297, 345, 350, 354
Step icons 345
stepping

simulator 280
source code 256
State Browser 351
State Diagram 350
state diagrams 274
states 267
steps
stepping 267
Steps Backward
State Diagram Window 243
Steps Backward Icon
Source Code Window 236
Steps Forward
State Diagram Window 243
Steps Forward Icon
Source Code Window 236
Stop Expansion icon 330
Stop icon 271
Stop Layout
Schematic Window 246
Stop Layout command 336
Stop Time field 171
stop_limit parameter 194
string equality operator 366
string operators 366
strings
searching 331
styles 217
Suite (see Undertow Suite)
supported simulator types 250
supported simulators 15, 31, 51, 255
sweep
parametric 221
Sync command 218
sync options 305
Sync Options dialog 305
Sync Panes command 122, 218
Sync Undertows command 73
synchronized windows 244
synchronizing
panes 122
plots 218
sessions 72
windows 306
zoom 218
synchronizing windows 72
Synopsys VCS 255, 491
syntax
filters 193
T
T0 cursor 61, 159, 209, 384, 391
T1 cursor 159, 209, 384, 391
T2 cursor 159
tab file 457
tabs
Hierarchy Location 296
Tag Anchors command 112
Tag icon 62
Tag Significant Digits field 99, 150
tags
measure 43, 113
text 43
title 111
value 43, 112
waveforms 110
Tags command 43, 110, 217
TAN 190
TANH 190
TANPI 190
tar 19
tar -xvf command 391
Target command 109
targets
cursors 109
tearing-off
windows 124
Template command 200, 376
Templates 198, 374
templates
Builder window 212, 387, 388
complete 213, 389
creating 204, 210, 379, 385
saving 205, 380
Templates list-box 206, 207, 382
Templates window 197, 200, 373, 376
Test Analyzer 125
Test Analyzer dialog 125
Test Analyzer window 126
TestAanalyzerSetTimingSignal function 371
testbench
FLI 464
PLI 456, 462, 468
text
adding 217

HDL windows 527
Text (Choose) command 78
Text icon 67, 295
text tag 43
text tags 43
text to send to simulator 308
Text widget 199, 200, 374, 375
Text window 87
threshold 229
tics 217
Tile Window command 314, 322
Tile Windows command 261, 266
time
cursors 55
scales 55
Time column
source code 323
time cursors 42, 61, 158, 209, 384, 391
time markers 53
Time pane 46, 90, 160, 180, 192, 195
time points
sequences 132
Time Scale command 120
Time Scale dialog 121
Time Scale window 120
time units
setting 120
time values
cursors 53
TIMEMILL 15, 31, 51
timing
falling edge 173
rising edge 172
Title field 75
Title Tag button 143
Title Tag command 217
title tags
waveforms 111
Title Tags command 111
titles 217
toggle
analysis 51
toggle analysis 31
Toggle Analyzer window 131
Tool bar 62
tool bar
Source Code Window 236
Toolbar 206, 329, 382
Builder window 211, 387
Toolbar command 211, 386
toolbars
analog icons 141
Chooser 66
chooser icons 146
icons 56
Schematic 329
source code 257
State 344
tools
analog 166
GoTo 50
Interactive 481
interactive 36
Interactive tool 70
Optimizing 481, 483
Persistence 281
PLI 481
Schematic Window 62
State Coincidence 281
State Diagram 62
State Table 281
Tools menu 125, 166, 323
Topline.script file 392
Trace command 300, 323
trace command 255
trace dump file 455, 485, 486
trace file 305
trace file, loading (source code viewer) 300
Trace files 455, 485
trace files 253, 254, 284
Trace Filter dialog 319
Trace Filter window 314
Trace Info File field 305
Trace window 353
Trace Window icon 354
tracefile 254, 284
traceFileName field 293
tracing 486
levels 487
netlists 31, 51
root 487
signals 341
tracing signals
schematic window 272
trailing

leading edge 297
trailing edge
searching 331
transforming
x axis 220
y axis 220
transient file 523
transitions
searching 297, 331
Trigger Conditions dialog 131
Trigger Functions 164
triggers
Logic Analyzer 131
Turn Animation On icon 354
Twos Compliment command 76
U
uncompress 19
Undertow
analog extensions 390
colors 522
commands 537
exiting 529
icon bar 53
installing 18
linking 535
main window 32, 52
menu bar 52
obtaining software 17
saving sessions 71
starting 32, 33, 34, 520, 525, 531, 535
starting new sessions 72
supported simulators 31
synchronizing sessions 72
using 31
Undertow command 518
undertow directory 516
Undertow Executable field 522
Undertow Host field 522
Undertow menu 517
Undertow Suite
description 231
overview iv, 231
starting 246, 282, 286, 291
windows 279
Undertow suite 256
batch operation 248
starting 252
Undertow window 52, 272, 280, 322, 518
Undertow window (see waveform window)
undertow.ile file 516
Undertow.Init file 515
Undertow.init file 517
UndertowAnalogExtension.pm module 397
UndertwoAnalogExtension.pm file 392
Undo Icon
Waveform Window 240
Undo icon 57
Unix
chmod 20
tar 19
uncompress 19
Unselect all Text button 528
until statement 364
unzip 19
Up command 336
Up Icon
Source Code Window 236
Up icon 330
Up icons 297
Update command 312, 336
Update Icon
Waveform Window 240
Update icon 60, 118
updating
signals 118
Upscope command 336
Use Analog Line Patterns button 98, 149
Use Glyphs button 337
Use HSpice Nomenclature button 98, 149
Use Probe File command 528
Use Project Name for Files button 488
Use Project Name For Files field 305
Use Schematic Name As Undertow Alias Name button 524
Use Text button 337
User Interface 217
using Undertow 31
ut command 34
ut executable 246
ut -iv command 248, 252, 253, 255, 284, 285, 286, 289, 292, 293, 456, 457, 458, 463, 469, 492
ut -v command 465, 469, 535, 536
UT_ROOT_DIR 21, 286, 456, 462, 499, 516

UT_ROOT_DIR directory 246
UT_ROOT_DIR environment variable 137, 456, 464, 468
UT_WORK_DIR 22, 516
UT_WORK_DIR environment variable 302
utConnectivity command 488
utLimitVCD command 139, 490
V
Value Alias window 66, 79, 87
Value Aliases field 49
Value area 54
value areas
  adding 123
  removing 123
Value button 123
Value Change Dump file (see VCD)
Value field 45
value labels 217
Value Tag dialog 113
Value Tag icon 62
value tags 43
  waveforms 112
Value Tags command 112
Value window
  configuring 87
Value/Glitch button 130
values
  absolute 229
  searching 297, 331
  signals 44, 54
  x-y 230
Values Significant Digits field 99, 150
Values Tag button 144
Variable button 269, 325
Variable dialog 213, 388
VariableGetIncrementalValue function 371
VariableGetIncrementalValueAfterCutoff function 397
VariableGetType function 371
VariableGetValues 396
VariableGetValues function 371
variables
  CDS_INST_DIR 468
  EDITOR 312
  environment (see environment variables)
  FINUSERPLIOBJ 507
  LPR_CALL 75
  ModelTech 464
  PATH 490
  path 456, 468
  PLI_BUILD 468
  PLIOBJS 463, 464, 465, 504
  scalars 362
  setting 203, 378
  UT_ROOT_DIR 137, 456, 464, 468
  UT_WORK_DIR 302
  VCSI_HOME 456
  ver_installdir 497
  widgets 200, 375
Variables button 280
Variables Icon
  Source Code Window 238
VariableSetValues function 371
vccdump program 507
vcd command 255
VCD file 69, 138, 283, 522, 533
VCD files 119, 254, 280
vconfig program 453, 454, 497, 498
VCS 491
  PLI 456
VCS simulator 284
vcsi command 457
VCSI_HOME environment variable 456
VCT branch 259
VCT multibranch 260
VCT report 300
VCT Report command 300
Vdump
  Fast files 137
vdump program 69, 119, 135, 137, 456, 468, 490
ver_installdir environment variable 497
verExecutable field 293
Verilog 15, 31, 51, 279, 526
  generating dump files 138
  generating Fast Files 139
  integration 530
  netlists 534
  simulation 522
Verilog executable
  creating 453
Verilog files 283
verilog window 526
Verilog XL simulator 284

verilog.dump file 138
Verilog-AMS Transient
  simulation 522
verilogI.ini file 517
Verilog-XL 255, 279, 455, 485, 497, 515, 517, 533
  static mode 498
Verilog-XL simulator 253
Veritools web site 491
veritools.com 491
VERITOOLS_LICENSE_FILE 19, 20, 22, 247
  environment variable 21
veriuser.c file 497
vertical
  scrolling 117
vertical grid
  pixels 101, 152
vertical panes 267, 349
VHDL 15, 31, 51
VHDL files 283
vhdl.dump file 465
vi editor 19, 22
View library 521
View Signal command 532
viewing
  memory 489
  memory contents 275
  modules 262
  output files 31, 51
  results 456, 457, 465, 469
  schematics 261, 266
  signals 38, 531
  state diagrams 266
  waveforms 37, 260, 294
views
  multiple 349
virtual simulation 260
voltage
  interpolating 132
Voltage pane 90, 160
Vsim 504
vsim program 465
VSS 15, 31, 51
vt_finsimuser.c file 507
vt_ptab file 507
vt_vcs.tab 492
vt_vcs.tab file 457
vt_veriuser.c file 453, 454, 497
VTCOMPRESS command 255, 455, 456, 457, 463, 465
VTCOMPRESS program 468
vtDump command 249, 490
vtDump file 457
vtDump program 119, 139, 287, 455, 456, 457, 462, 463, 468, 490
vtdump program 119
vtDumpMem command 489
vtDumpMemAddr command 489
vtDumpnames 484
VtDumpvars
  parameters 482
vtDumpvars command 249, 255
vtDumpvars program 139, 287, 454, 456, 457, 463, 468, 482
vtpli.o file 453, 454, 497
vtplifinsim.o file 507
vtplimodtech.so file 504
vtplivcs.o file 457
vtrun program 455, 485
vtrun -trace command 455, 486
vtrun -tracefile command 455, 486
vtrun.ascii file 486
vtrun.ps file 486
vtTrace command 486
vtTracename command 487
W
wav files 284
Wave Window menu 98, 149
Wave-Display command 318
Waveform Background Color field 97, 148
Waveform Chooser window 187
Waveform command 300
Waveform Display Area 241
Waveform Height command 41, 160
Waveform Heights command 217
WaveForm Heights dialog 122
Waveform Spacing field 98, 149
Waveform Starting Offset field 98, 149
Waveform Viewer 302
Waveform viewer 270, 489
waveform viewer 322, 351
Waveform Window

Copy Icon 240
Cut Icon 240
GoTo Icon 240
Group Icon 240
Help Icon 241
Paste Blank Icon 240
Paste Icon 240
Scroll To Icon 240
Select All Icon 240
Signal Name Area 241
Signal Values Area 241
Snap Icon 240
Snap Source Icon 240
Undo Icon 240
Update Icon 240
Waveform Display Area 241
Zoom Full Icon 240, 243
Zoom In Icon 240
Zoom Out Icon 240
Waveform window 66, 117, 209, 245, 256, 261, 273, 280, 281, 286, 384, 489
waveform window 239
Waveform Window Choose Icon 240
waveforms
analog 220, 228
annotating 110
averaging 133
bundling 83
creating 219
displaying 31, 39, 55, 260
displaying source code 270
editing 219
fit to pane 113
height 41, 122
measure tags 113
overlaying 132, 147
panning 41
plotting 214
slopes 133
snapping to source code 261
spacing 91
synchronizing 122
tags 110
title tags 111
value tags 112
viewing 37, 260, 294
zooming 40
waveforms - displaying 300
web
obtaining software 17
web site
Veritools 491
while statements 363
Widgets 198, 374
widgets
adding 212, 388
Buttons 198, 374
configuring 199, 375
Current Value 208, 383
destroying 200, 375
Entry 198, 203, 374, 378
flashing 200, 375
Frame 198, 374
Label 198, 202, 374, 377
List Box 198, 374
Radio Button 199, 374
Scroll Bar 199, 374
Text 199, 200, 374, 375
variables 200, 375
Widgets list-box 206, 207, 382, 383
widgets window 198, 374
width
pulses 175
window
Signal Browser 67
Window menu 96, 134, 178, 313, 337, 349
windows
activating 208, 383
Alter Signal 313, 314
Analog Expression 179, 191, 192
Analog Filter Builder 178
Analog Functions 178, 181, 187
Analog To Digital 170
antrim 526
Boolean Functions 134, 176, 181
Break Point 314
Breakpoint 315
breakpoints 323
Builder 197, 198, 201, 208, 373, 377, 383, 384
Bundle 94
Buttons Definition 307
Choose 272, 277
Chooser 275, 489
Connectivity Analyzer 127
Connectivity History 128
creating 201, 208, 377, 383
design files 464
Detail 312

Display 52, 55, 127
Enable Results 261
Encapsulation 517, 530
Encapsulation Control 534
Error 372
Execution Marks 324
Eye Diagram Signals 133, 171
Find Logic 130
floating 124
Floating Groups 63
HDL text 527
icons 124
Inspect Signal 314
Logic Analyzer 128
main 32, 52
menus 58, 124
Modify Group 64
module list 58
New Command 199, 374
New Text 200, 375
open 107
Output 94, 162
Output Display 275
Overlay Analog Signals 132, 163, 168
Pane Controller 90
Pane Zoom 91
Results 202, 205, 207, 309, 314, 377, 380, 383
Schematic 128, 261, 263, 266, 272, 273, 281, 327, 336, 342, 343, 531, 532
schematic 244, 271, 272
Script Tool 183, 196, 202, 210, 361, 377, 386
Search 314
Select Overlay Properties 181
Selected Analog Waveform 182
Selected Signals 79, 80, 87
Show Connectivity 128
Show Selector 338
Signal List 79
signal list 58
Signal Name 45
Signal Selector 245, 260, 273
Signals Found 296
Simulator Executable 250, 251
Simulator Output Options 304
size 160
Source 258, 324, 328, 464, 489
Source Code 66, 235, 243, 245, 256, 261, 270, 272, 273, 280, 281, 286, 294, 301, 302, 309, 312, 313, 319, 343, 344, 352, 405, 440
source code 235, 295
Source Code Files/Simulator Options 303
Source Code Window 235
Source Window 233
State 261, 266, 267
State Diagram 266, 274, 280, 344, 350
state diagram 242
State Viewing 346
synchronized 244
synchronizing 72, 306
Templates 197, 200, 373, 376
Test Analyzer 125, 126
Text 87
tiling 322
Time Scale 120
Toggle Analyzer 131
Trace 353
Trace Filter 314
Undertow 52, 272, 280, 322, 518
Undertow Suite 279
Value 87
Value Alias 66, 79, 87
verilog 526
Waveform 66, 117, 209, 245, 256, 261, 273, 280, 281, 286, 384, 489
waveform 239
Waveform Chooser 187
widgets 198, 374
Windows menu 45, 126
wlf files 284
Working Dir command 87
Working Directory command 178
Working Directory field 126, 337
Working Directory filed 134
WRITE 185
www.veritools.com 491
WYSWYG printing 74
X
X 186
x axis
transforming 220
Xdefaults file 218, 372
XLIN 186
XLOG 186
xrdb program 217
XVAR 186

x-y values 230
Y
Y 186
y axis
  transforming 220
Y-Axis Scale Significant Digits field 99, 150
YLIN 186
YLOG 186
YRANGE 186
Z
zoom
  buttons 40
  full 40
  in 40
  mouse 40
  Normalize 161
  out 40
  synchronizing 218
Zoom 1 to 1 command 348
Zoom 1 to 1 Icon
  Schematic Window 243, 245
Zoom Full command 336, 348
Zoom Full Icon
  Schematic Window 245
  Waveform Window 240, 243
Zoom Full icon 59, 330, 344
Zoom function 371
Zoom In command 336, 348
Zoom In Icon
  Schematic Window 245
  State Diagram Window 243
  Waveform Window 240
Zoom In icon 59, 330, 344
Zoom In, Both button 141
Zoom In, Vertical button 141
Zoom One to One icon 330, 344
Zoom Out command 336, 348
Zoom Out Icon
  Schematic Window 245
  State Diagram Window 243
  Waveform Window 240
Zoom Out icon 59, 330, 344
Zoom Out, Both button 141
Zoom Out, Vertical button 141
Zoom Time command 168
zooming 159
  panes 90